
UNIVERSIDADE FEDERAL DO AMAZONAS - UFAM

FACULDADE DE TECNOLOGIA - FT

# SISTEMA DE GERENCIAMENTO DE ASSET ADMINISTRATION SHELLS NO CENÁRIO DA INDÚSTRIA 4.0

Evaldo Patrik Dos Santos Cardoso

Manaus - AM

Outubro de 2023

Evaldo Patrik Dos Santos Cardoso

# SISTEMA DE GERENCIAMENTO DE ASSET ADMINISTRATION SHELLS NO CENÁRIO DA INDÚSTRIA 4.0

Monografia de Graduação apresentada à Coordenação do Curso de Engenharia da Computação da Universidade Federal do Amazonas, UFAM, como parte dos requisitos necessários à obtenção do título de Engenheiro da Computação.

Orientador(a)

Vicente Ferreira de Lucena Junior, Dr.

Universidade Federal do Amazonas - UFAM

Faculdade de Tecnologia - FT

Manaus - AM

Outubro de 2023

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

Cardoso, Evaldo Patrik Dos Santos
C268s Sistema de Gerenciamento de Asset Administration Shells no Cenário da Indústria 4.0 / Evaldo Patrik Dos Santos Cardoso . 2023
109 f.: il. color; 31 cm.

Orientador: Vicente Ferreira de Lucena Junior
TCC de Graduação (Engenharia da Computação) - Universidade Federal do Amazonas.

1. Gerenciamento de Shell. 2. Asset Administration Shell. 3. Indústria 4.0. 4. Interoperabilidade. I. Lucena Junior, Vicente Ferreira de. II. Universidade Federal do Amazonas III. Título

Monografia de Graduação sob o título *Sistema de Gerenciamento de Asset Administration Shells no Cenário da Indústria 4.0* apresentada por Evaldo Patrik dos Santos Cardoso e aceita pela Faculdade de Tecnologia da Universidade Federal do Amazonas, sendo aprovada por todos os membros da banca examinadora abaixo especificada:

Prof. Vicente Ferreira de Lucena Junior, Dr
Orientador(a)
Departamento de Eletrônica e Computação (DTEC)
Universidade Federal do Amazonas

Prof. Thiago Brito Bezerra, Dr
Departamento de Eletrônica e Computação (DTEC)
Universidade Federal do Amazonas

Prof. Renan Landau Paiva de Medeiros, Dr
Departamento de Eletricidade (DE)
Universidade Federal do Amazonas

Manaus - AM, 10 de novembro de 2023.

Dedico este trabalho a duas fontes inestimáveis de inspiração em minha vida. Aos meus avós paternos, cuja sabedoria e amor continuam a guiar-me, mesmo na ausência física. Sua memória é uma bênção que ilumina meu caminho a cada passo. Aos meus avós maternos, cujo apoio constante e encorajamento foram fundamentais para esta conquista. Suas palavras de incentivo e presença constante foram um farol de luz ao longo desta jornada acadêmica. A todos vocês, minha gratidão é imensa. Este trabalho é dedicado em reconhecimento ao papel singular que cada um de vocês desempenhou em minha trajetória.

# AGRADECIMENTOS

À Jesus Cristo, meu Salvador e Fonte de Força ao longo desta jornada acadêmica. Sua orientação e amor incondicional foram minha luz nos momentos de desafio e incerteza, em meio aos desafios e às dificuldades, encontrei conforto na promessa de que todas as coisas cooperam para o bem daqueles que O amam.

À meus pais, Edvaldo Cardoso e Edileuza Cardoso, agradeço por serem meus pilares de força. Nunca mediram esforços para que eu fosse à universidade, suas palavras de incentivo e confiança em mim foram a motivação que me impulsionou a alcançar este objetivo. Seu constante apoio emocional e suporte prático foram fundamentais. Vocês merecem esse momento tanto quanto eu e quero compartilhá-lo, eu os amo incodicionalmente.

À meu irmão amado Renan Cardoso, agradeço por ser meu parceiro e o apoio incondicional foram uma fonte de inspiração. Compartilhar este desafio com você fortaleceu nosso vínculo e enriqueceu minha experiência. Que eu seja inspiração para você também conquistar o seu título de engenheiro.

À minha irmã amada Eduarda Santos, futura psicóloga. Suas palavras de encorajamento e confiança em mim foram o impulso que me guiou até aqui. Sua presença constante e apoio emocional foram pilares essenciais ao longo deste caminho.

À todos da minha família, que estiveram presentes, que me apoiaram, aconselharam e incentivaram durante esta longa jornada, muito obrigado.

À meus adoráveis amigos Luís Mota, Leandro Maciel, Carla Vieira, Jimmy Villalaz, Lucas Castro. Neste momento de conquista, quero expressar meu profundo agradecimento por ter amigos tão maravilhosos em minha vida. Vocês são uma parte

essencial desta jornada e o meu coração se enche de gratidão por tê-los ao meu lado.

À meu orientador, Dr. Vicente Ferreira de Lucena Junior, pelo seu valioso apoio e orientação ao longo desta jornada acadêmica. Além disso, quero agradecer por sua paciência e disposição em partilhar seu conhecimento, sempre com um espírito de colaboração e incentivo ao meu desenvolvimento acadêmico.

À todas as boas amizades criadas durante a minha jornada acadêmica. Além disso, aos meus colegas e veteranos do Bug Hackers por partilharem as experiências e conhecimentos diversos, e à todos que tiveram influência direta e indiretamente na minha formação.

E por fim, um especial agradecimento à meus amados colegas do time de software do Laboratório de Sistema Embarcados (LSE): Isaque Vilson, Alison Almeida, Heitor Lifsitch, Vitor Norton, João Furquim, Maicon Pantoja, Gabriella Rabelo, Lohan Victor, Nádila Azevedo, Gabrielle Mestrinho, Giovana Teodoro e Fábio Wendel, que são pessoas incríveis, que estiveram prómixos, e colaboraram significativamente no desenvolvimento deste projeto, passamos por desafios durante essa caminhada, discussões acaloradas, noites mal dormidas e muitos viradôes de código, contudo, como diria Allan Amorin: "Quem disse que seria fácil?".

Com muita gratidão e respeito, Evaldo Cardoso.

*"Tudo o que fizerem, façam de todo o coração, como para o Senhor, e não para os homens."*

Bíblia, Cl 3:23

# SISTEMA DE GERENCIAMENTO DE ASSET ADMINISTRATION SHELLS NO CENÁRIO DA INDÚSTRIA 4.0

Autor: Evaldo Patrik Dos Santos Cardoso

Orientador: Vicente Ferreira de Lucena Junior, Dr.

## Resumo

O presente trabalho aborda o tema de gerenciamento de Asset Administration Shells (AAS) no contexto da Indústria 4.0. Os AASs são uma representação digital detalhada de ativos físicos, fornecendo uma base para uma gestão de ativos mais eficiente e inteligente. É demonstrado que a implementação bem-sucedida dos AASs oferece benefícios significativos, incluindo a capacidade de monitoramento em tempo real, eventos, logs e outros. Além disso, a padronização e interoperabilidade promovem a integração eficaz em ambientes industriais complexos, permitindo a colaboração entre diferentes ativos e sistemas, independentemente do fornecedor. O gerenciamento de Shells na Indústria 4.0 não é apenas uma estratégia de eficiência operacional, mas uma necessidade imperativa para organizações que buscam prosperar na era da transformação digital. No geral, este trabalho serve como um ponto de partida para uma jornada contínua em direção a uma gestão de ativos mais eficaz e resiliente na era da Indústria 4.0 visando inspirar iniciativas futuras e promover uma reflexão mais ampla sobre o potencial da gestão de shells.

*Palavras-chave*: Gerenciamento de Shell; Asset Administration Shell; Indústria 4.0; Interoperabilidade;

# SISTEMA DE GERENCIAMENTO DE ASSET ADMINISTRATION SHELLS NO CENÁRIO DA INDÚSTRIA 4.0

Autor: Evaldo Patrik Dos Santos Cardoso

Orientador: Vicente Ferreira de Lucena Junior, Dr.

## Abstract

The present study addresses the topic of Asset Administration Shells (AAS) management in the context of Industry 4.0. AASs are a detailed digital representation of physical assets, providing a foundation for more efficient and intelligent asset management. It is demonstrated that the successful implementation of AASs offers significant benefits, including real-time monitoring, events, logs, and more. Furthermore, standardization and interoperability promote effective integration in complex industrial environments, enabling collaboration among different assets and systems, regardless of the provider. Managing Shells in Industry 4.0 is not just an operational efficiency strategy, but an imperative need for organizations looking to thrive in the age of digital transformation. Overall, this work serves as a starting point for more effective, resilient asset management in the era of Industry 4.0. It aims to inspire future initiatives and promote broader reflection on the potential of shell management.

*Keywords*: Shell Management; Asset Administration Shell; Industry 4.0; Interoperability;

# LISTA DE ILUSTRAÇÕES

# LISTA DE TABELAS

# Lista de abreviaturas e siglas

***AAS*** *Asset Administration Shell*

***AI*** *Artificial Intelligence*

***AM*** *Additive Manufacturing*

***API*** *Application Programming Interface*

***CLI*** *Command-Line Interface*

***CPS*** *Cyber-physical Systems*

***DT*** *Digital Twin*

***GTAI*** *Germay Trade & Invest*

***HTTP*** *Hypertext Transfer Protocol*

***I4.0*** *Indústria 4.0*

***IDE*** *Integrated Development Environment*

***IoT*** *Internet of Things*

***JSON*** *JavaScript Object Notation*

***LDS*** *Local Discovery Service*

***OPC UA*** *OPC Unified Architecture*

***PIM*** *Polo Industrial de Manaus*

***RF*** *Requisito Funcional*

***TS*** *TypeScript*

***VS Code*** *Visual Studio Code*

# SUMÁRIO

# 1

# INTRODUÇÃO

A *Indústria 4.0* (*I4.0*) é um termo criado pela *Germay Trade & Invest* (*GTAI*), agência de desenvolvimento econômico da República Federal da Alemanha, nomeada para promover a ideia de que estamos na origem de uma nova revolução industrial. Devido ao aparecimento, avanço e convergência de uma série de tecnologias isso permite uma ligação quase em tempo real entre os reinos físico e digital. Este casamento físico-digital é impulsionado por: manufatura aditiva (*AM*); a Internet das coisas (*IoT*); a cadeia de bloqueio; a robótica avançada; a inteligência artificial (*AI*) e outras tecnologias relacionadas (OLSEN; TOMLIN, 2020).

De acordo com (Plattform Industrie 4.0, 2022), a *I4.0* refere-se à rede inteligente de máquinas e processos para a indústria com a ajuda da tecnologia da informação e comunicação. Nomeada de 4ª revolução industrial, tem como fundamento a introdução de dispositivos dotados de relativa inteligência com capacidade de interação autônoma e direta ao longo da cadeia de valor da empresa. Essa capacidade pode ser provida pelo desenvolvimento de sistemas Físico-Cibernéticos (*Cyber-physical Systems - CPSs*), que são sistemas compostos por entidades computacionais colaborativas visando o controle de entidades físicas.

Os *CPSs* possuem capacidade de auto-organização, de monitoramento de processos e criam representações do mundo real no ambiente computacional ou virtual. O desenvolvimento de sistemas físico-cibernéticos permitirá a incorporação efetiva do paradigma da *I4.0* nas empresas através da possibilidade de introdução dos conceitos e de tecnologias como: *IoT* (Internet das Coisas; do inglês *Internet of Things*); de Big

Data área que estuda como tratar, analisar e obter informações a partir de um conjunto grande de dados; Cloud Computing (computação em nuvem) e de inteligência artificial.

Nesse mesmo contexto os objetos físicos denominados Assets, necessitam de conexão com um ambiente virtual. Comumente denominado Shell o Asset Administration Shell (*AAS*) é a representação digital do Asset responsável pela conexão desses objetos físicos nesse contexto (Plattform Industrie 4.0, 2022). Os *AASs* armazenam as informações e os dados sobre os objetos físicos e também atuam como uma *API* de comunicação padronizada disponível para integrar seus dados. Inicialmente cada Asset necessita ter seu próprio Shell que permita sua integração com a *I4.0*.

## 1.1 Contextualização

Com a acelerada evolução da tecnologia de informação e dos processos industriais no *Polo Industrial de Manaus* (*PIM*), algumas empresas tentam acompanhar essa iminente transformação da indústria tradicional para um modelo de indústria globalmente competitivo.

O gémeo digital (no inglês, *Digital Twin*s — *DT*)) é uma representação digital abrangente de um produto individual que irá desempenhar um papel integral num ciclo de vida do produto totalmente digitalizado. Neste contexto, o gémeo digital pode ajudar a assegurar continuidade da informação ao longo de todo o ciclo de vida, comissionamento virtual de sistemas, e apoio à decisão e previsões de comportamento do sistema na fase de desenvolvimento, bem como em todas as fases subsequentes do ciclo de vida (HAAG; ANDERL, 2018).

Segundo (STOJANOVIC et al., 2021), existe uma procura crescente na produção industrial para permitir a sustentabilidade e a resiliência nas redes da cadeia de abastecimento, o que significa que os ecossistemas *DTs* se estendem por múltiplas organizações. A fim de evitar depender de ambientes proprietários baseados na nuvem, iniciativas e tecnologias estão a sendo desenvolvidas para manter o controle sobre a utilização de *DT*s. As futuras arquiteturas e ferramentas de engenharia e gestão de *DTs* serão posicionadas em tais ambientes.

Com isso, os *Asset Administration Shell*s ajudam a implementar gêmeos digitais no cenário da *Indústria 4.0* e criar integrações entre as soluções de diferentes fornecedores. Porém, o processo desde a modelagem até a implantação (deployment) e manutenção desses *DTs* tornam o trabalho cansativo, a medida que o número de Shells cresce.

Notou-se então que era necessário modelar e prototipar soluções para gerenciar e monitorar o ciclo de vida dos Shells na conjuntura da *Indústria 4.0* desde a criação do seu arquivo de modelagem até a homologação do modelo virtual criado.

## 1.2 Justificativa

A atual conjuntura pós quarta revolução industrial transformou a forma como acontece o gerenciamento de ativos, e em como estas operam os mesmos. O *AAS* passa a ser nesse contexto, uma peça fundamental para garantir a eficiência operacional e a competitividade das organizações.

Como os ambientes industriais modernos demandam uma abordagem mais inteligente e integrada dos seus ativos, o que foi citado anteriormente, o uso do *AAS* traz uma representação digital detalhada e padronizada desses ativos, gerando a possibilidade de uma gestão mais eficaz e permitindo o monitoramento em tempo real, análises preditivas e uma manutenção proativa dos mesmos.

Este estudo, então, surge diante da necessidade de otimizar tais processos industriais, reduzir custos operacionais e busca aumentar a produtividade. Implementar um Sistema de gerenciamento *AAS* não apenas possibilita o acesso a informações valiosas sobre o estado e desempenho dos ativos, mas também promove a integração entre diferentes sistemas e a interoperabilidade entre ativos de diferentes fornecedores.

Além disso, o sistema proposto é uma iniciativa estratégica que não apenas atende às demandas atuais, mas também posiciona a organização para enfrentar os desafios futuros da Indústria 4.0. Ao adotar uma abordagem proativa para a gestão de ativos, as empresas podem ganhar uma vantagem competitiva significativa em um cenário industrial em constante evolução.

## 1.3 Objetivos

### 1.3.1 Objetivo Geral

O objetivo deste trabalho é modelar e prototipar de um sistema constituído por módulos de software, que operam de forma sistêmica viabilizando o gerenciamento de Shells, seguindo o modelo de referência para arquitetura da *Indústria 4.0*. Um sistema de gerenciamento de *AAS*s capaz de gerenciar o ciclo de vida de Shells na arquitetura de software proposta, para qualquer empresa que utilize processos físicos-cibernéticos.

### 1.3.2 Objetivo Especifico

Desenvolver um sistema com ambiente de criação de Shells, instalação e monitoramento, permitindo certo nível de automatização no processo de criação dessas instâncias. O sistema terá arquitetura distribuída, com interfaces padronizadas que proporcione integração, uso transparente, modularidade, flexibilidade e escalabilidade.

## 1.4 Organização do Trabalho

Este trabalho segue uma estrutura organizada em cinco capítulos, uma seção de referências e cinco apêndices, proporcionando uma abordagem detalhada para a pesquisa em questão.

No Capítulo 1, a introdução estabelece a base do estudo, descrevendo a justificativa por trás do sistema, contextualizando o tema e definindo objetivos gerais e específicos, o que serve como alicerce para as seções subsequentes.

Já no Capítulo 2, as ferramentas e tecnologias utilizadas no projeto são minuciosamente exploradas, abrangendo elementos variados, como Asset Administration Shell, TypeScript e Docker, destacando a base tecnológica do trabalho.

O Capítulo 3 concentra-se no Gerenciador de Shells, seus requisitos funcionais, com exemplos de casos de uso que ilustram sua aplicação prática e sua arquitetura.

Os resultados alcançados durante a pesquisa são apresentados no capítulo 4.

Isso inclui a criação e modelagem de Shells, o processo de deploy e atualização, bem como detalhes sobre o gerenciamento de Shells, os testes realizados e as configurações específicas adotadas.

Por fim, o Capítulo 5 encerra o trabalho, oferecendo espaço para discussão dos resultados obtidos e lições aprendidas ao longo da pesquisa. Além disso, esse capítulo fornece uma perspectiva futura, sugerindo direções possíveis para trabalhos subsequentes.

O conjunto é enriquecido por cinco apêndices, que incluem, um documento de modelagem, um documento de requisitos, um documento de caso de uso, um plano de projeto, e um documento de testes. Esses apêndices fornecem informações adicionais relevantes e detalhadas do trabalho.

# 2

# FERRAMENTAS E TECNOLOGIAS UTILIZADAS

## 2.1 Asset Administration Shell

O *Asset Administration Shell* (*AAS*) é a representação digital de um bem. O *AAS* é composto por uma série de submodelos nos quais existem todas as informações e funcionalidades de um determinado ativo - incluindo as suas características, características, propriedades, estados, parâmetros, dados de medição e capacidades podem ser descritas (Plattform Industrie 4.0, 2022). Além disso, permite a utilização de diferentes tipos de comunicação e aplicações e serve como a ligação entre o objeto e a sua representação digital.

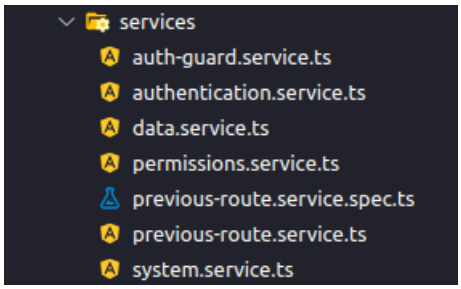
Figura 1 – Representação de um *AAS* e seu Asset


Fonte: (PLATTFORM INDUSTRIE 4.0, 2022)

## 2.2 TypeScript

*TypeScript* (*TS*) é uma linguagem de programação fortemente tipada que se baseia no JavaScript, oferecendo melhores ferramentas na hora do desenvolvimento (TypeScript, 2022). Utilizou-se desta linguagem para a codificação das interfaces de usuário, criação de serviços de requisições *HTTP* (Protocolo de Transferência de Hipertexto, do inglês, *Hypertext Transfer Protocol*), implementação de componentes específicos para o sistema que não estavam disponíveis na biblioteca de componentes de interface do usuário. A figura 2 exemplifica alguns arquivos de serviços implemetados no projeto, note que os arquivos de que usam *TypeScript* possuem *.ts* como extensão.

Figura 2 – Arquivos de serviços implementados em *TypeScript*




Fonte: (Elaborado pelo autor, 2023)

## 2.3 Angular

Angular é uma plataforma de desenvolvimento, construída em TypeScript que possui uma estrutura baseada em componentes para a criação de aplicativos da Web escaláveis, uma coleção de bibliotecas bem integradas que cobrem uma ampla variedade de recursos, incluindo roteamento, gerenciamento de formulários, comunicação cliente-servidor, além disso, contêm ferramentas para ajudá-lo a desenvolver, criar, testar e atualizar códigos(Angular, 2022).

Neste Projeto utilizou-se Angular CLI (*Command-Line Interface*) na versão 10+ como é ilustrado na figura 3, pois apresentava maior estabilidade de suas funcionalidades, maior compatibilidade com as demais tecnologias utilizadas neste projeto.

Além disso, esta plataforma possibilita a criação de componentes e códigos reutilizáveis viabilizando o desenvolvimento rápido e manutenção efetiva.

Figura 3 – Angular CLI versão 10+


Fonte: (Elaborado pelo autor, 2023)

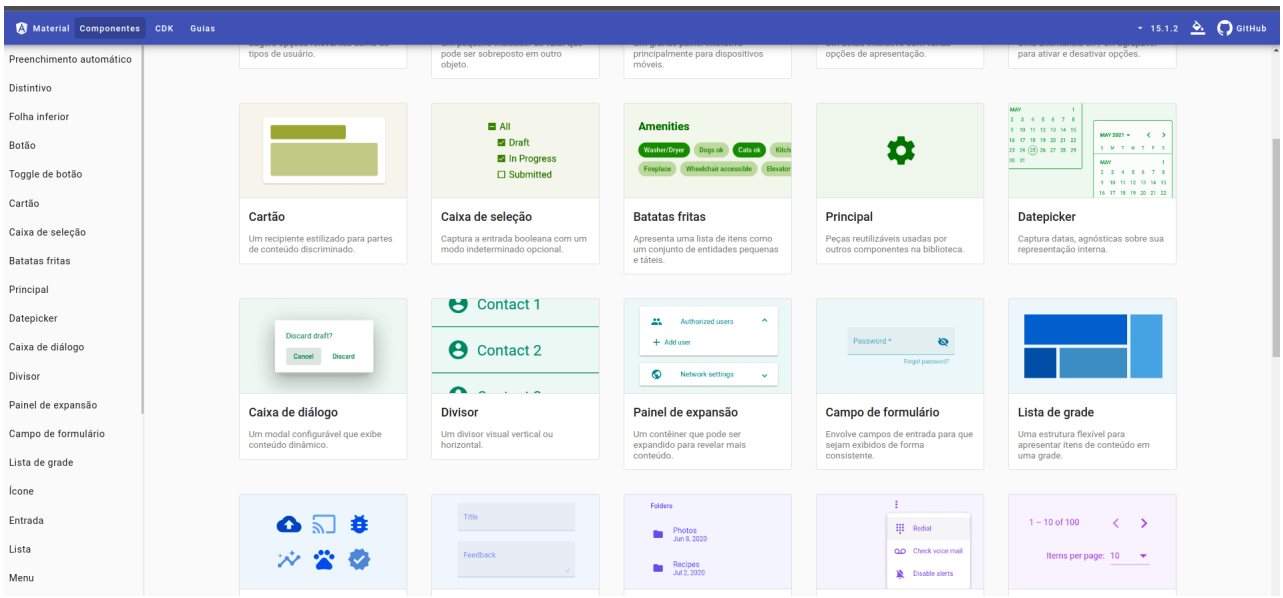
## 2.4 Angular Material Design

Angular Material Design é uma biblioteca de componentes de interface do usuário e componentes de design para aplicações criadas com Angular compatíveis com sistemas, web e desktop.

Utilizou-se esta biblioteca de componentes afim de facilitar o desenvolvimento das interfaces do sistema por ter muitos componentes já implementadas, isso agilizaria a customização e criação de uma identidade visual única para a aplicação de forma que a disposição dos elementos nas telas desenvolvidas fossem intuitivos e fluídos, proporcionando uma melhor experiência para os usuários.

Alguns dos componentes disponíveis na biblioteca estão listados na documentação e podem ser observados na figura 4 , como, por exemplo, cartões, modais (caixas de diálogo), ícones e formulários.

Figura 4 – Grade de componentes - Material Design

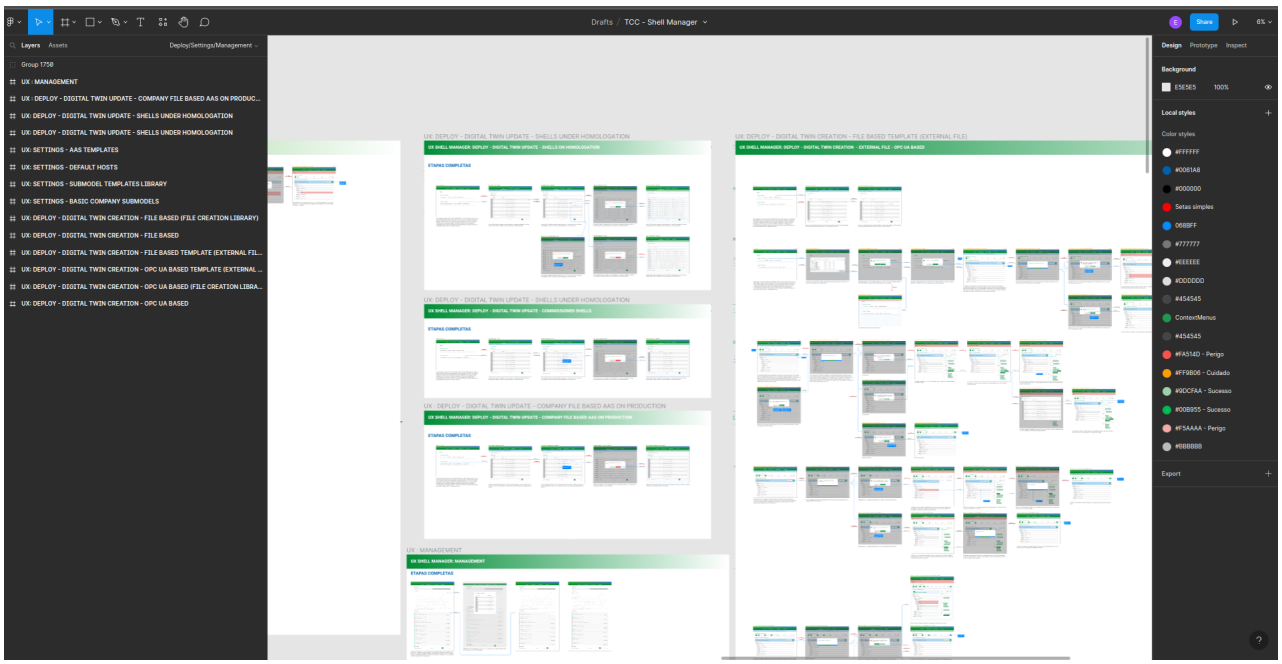



Fonte: (Elaborado pelo autor, 2023)

## 2.5 Figma

Figma é uma plataforma web de construção de interfaces de usuário e prototipação de fluxo de sistemas. Utilizou-se desta ferramenta para realizar a criação de toda a identidade visual do projeto, componentes, regra de cores e tudo o que envolve o design do sistema como ilustrado na figura 5. Foi escolhida por auxiliar na codificação dos elementos das interfaces, possuir características implementadas que possibilitam do desenvolvedor de inspecionar os elementos que estão sendo codificados, informando as propriedades relevantes que seriam adicionadas ao código, como, por exemplo, cor, altura, comprimento, formatação, tipo de fonte, entre outras.

Figura 5 – Interface de edição das telas no Figma

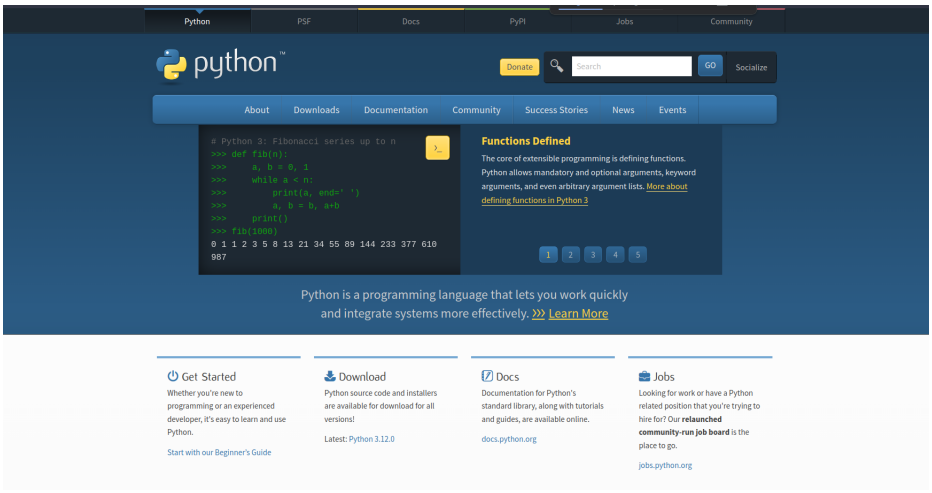


Fonte: (Elaborado pelo autor, 2023)

## 2.6 Python

Python é uma linguagem de programação de alto nível interpretada, orientada a objetos com semântica dinâmica. Seu alto nível de dados incorporados estruturas, combinadas com tipagem dinâmica e ligação dinâmica, fazem isso ser atraente para o Desenvolvimento Rápido de Aplicações, bem como para uso como uma linguagem de script ou para conectar componentes existentes. Python suporta módulos e pacotes, o que incentiva o programa modularidade e reutilização de código. (Python Software Foundation, 2023).

Figura 6 – Página inicial da Python Foudation




Fonte: (Python Software Foundation, 2023)

## 2.7 MongoDB - Banco de Dados

O MongoDB ilustrado pela figura 7, é um banco de dados de documentos projetado para facilitar a aplicação desenvolvimento e escala (MongoDB, 2023).

O formato do documento é baseado no *JSON* (*JavaScript Object Notation*), um esquema popular de armazenamento de estruturas de dados arbitrárias, além disso é conhecido por sua grande performace, disponibilidade e por baixo custo devido a quatro principais características: orientado a documentos, extremamente extensível, horizontalmente escalável, possuir muitos drivers (HOBERMAN, 2014).

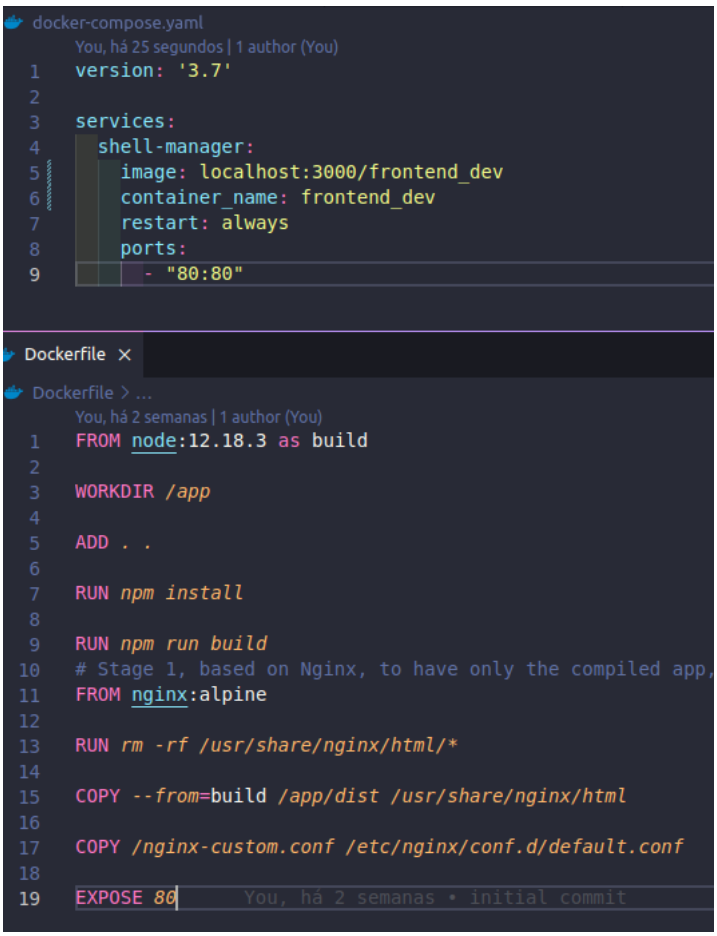
Figura 7 – Arquivo de configuração do contêiner front-end


Fonte: (MongoDB, 2023)

## 2.8 Docker

Utilizou-se Docker para facilitar a criação e administração de ambientes isolados chamados contêineres virtuais. Essa plataforma possibilitou a separação dos ambientes dos *Asset Administration Shell*s garantindo que não haja interferência entre as gêmeos digitais e seus respectivos dados. Além disso, permite que o sistema seja escalável, podendo ser replicado, modularizado ou até mesmo ser centralizado em um único ambiente. Na figura 8 podemos visualizar os arquivo de configuração do ambiente virtual do módulo de front-end.

Figura 8 – Arquivo de configuração do contêiner front-end

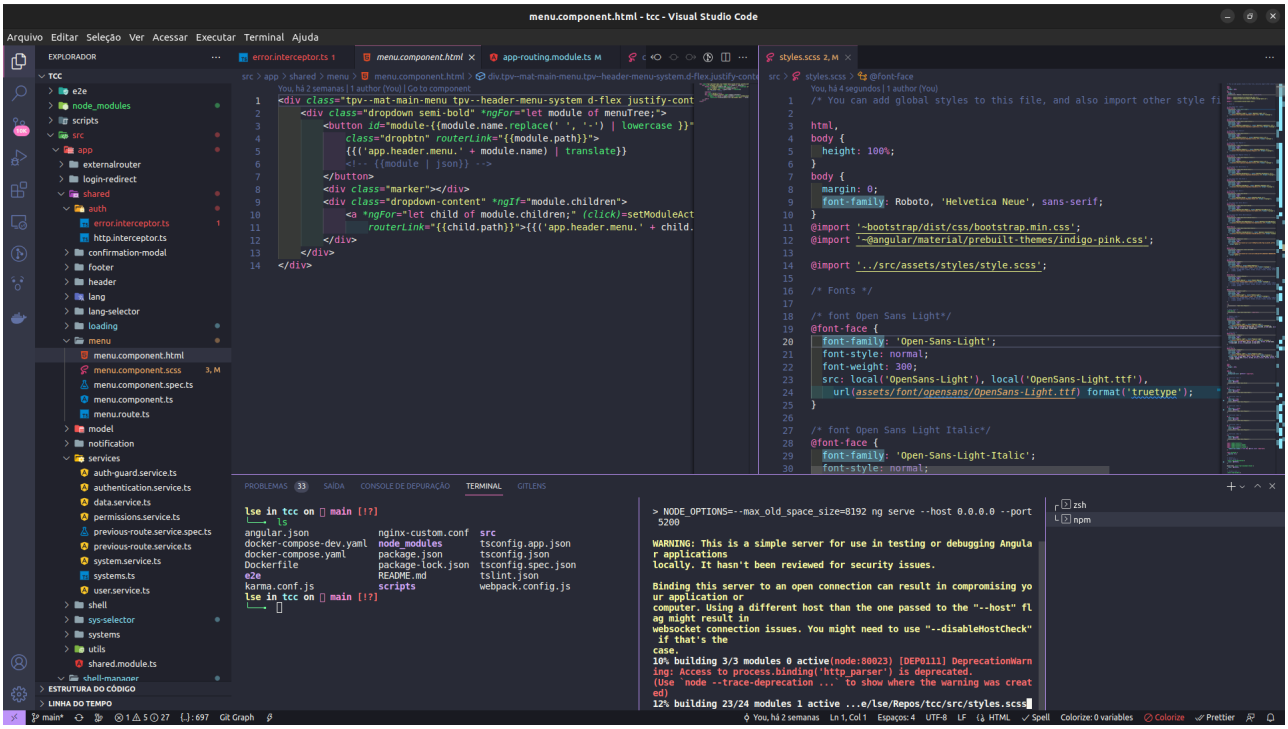



Fonte: (Elaborado pelo autor, 2023)

## 2.9 Visual Studio Code

*Visual Studio Code* (*VS Code*) é uma *IDE* (*Integrated Development Environment*) de código optimizado para a construção, depuração de aplicações web, criada pela empresa microsoft. Este editor de códigos ilustrado na figura 9 foi utilizado para a codificação das interfaces de cada módulo do sistema de gerenciamento de Shells. O *VS Code* mostrou-se uma ótima ferramenta para desenvolvimento deste projeto, apesar de existirem *IDEs* similares no mercado, pois possui integração com diversos plugins de acesso remoto, de versionamento e análise de código, além disso, possibilita visualização de múltiplos arquivos de código, terminais de linha de comando, facilitando o gerenciamento, edição de arquivos, e criação de servidores de teste e desenvolvimento.

Figura 9 – Interface do editor *VS Code*




Fonte: (Elaborado pelo autor, 2023)

# 3

# GERENCIADOR DE SHELLS

Neste catítulo são descritos os principais elementos que compuseram o processo de criação do software e funcionalidades que foram empregados nesse projeto. Estes conhecimentos foram eleitos com base na disciplina de introdução de engenharia de software, onde foi aluno. Como isso, foi explorado três áreas cruciais: requisitos, casos de uso e arquitetura.

Começaremos com os requisitos do sistema, onde foi usado para elencar, analisar e documentar as necessidades do cliente. Entender o que o cliente realmente desejava foi primeiro passo para o sucesso do desenvolvimento desse projeto.

Em seguida, adentraremos nos casos de uso, uma ferramenta essencial para compreender as interações entre os usuários e o sistema. Isso permitiu capturar de forma precisa e abrangente os cenários de uso essenciais para direcionar o desenvolvimento do sistema.

Por fim, a arquitetura de software, onde examinamos a estrutura e padrões que forneceram a base para uma aplicação robusta sendo possível compreender como projetar um sistema escalável e eficaz, o que é crucial para o sucesso do gerenciamento de Shells.

## 3.1 Requisitos do Sistema

### 3.1.1 Requisitos funcionais

Na tabela 1, são apresentados os requisitos funcionais do sistema de gerenciamento de Shells no contexto da *Indústria 4.0*. Os requisitos foram classificados em: Essencial (ESS), Importante (IMP), Desejável (DES).

Tabela 1 – Lista de requisitos funcionais

| CÓDIGO | IDENTIFICAÇÃO | ESS | IMP | DES |
|---|---|---|---|---|
| | **Modelagem Arquitetural** | | | |
| GS-SW-10 | Modelagem do Sistema | x | | |
| GS-SW-20 | Prototipação do Sistema (UI/UX) | x | | |
| | **Shells** | | | |
| GS-SW-30 | Criação de Shells a partir de uma base de dados | | x | |
| GS-SW-40 | Criação de Shells do tipo arquivo | x | | |
| GS-SW-50 | Criação de Shells Passivos | x | | |
| GS-SW-60 | Criação de Shells Ativos | x | | |
| GS-SW-70 | Edição de Shells template | x | | |
| GS-SW-80 | Biblioteca de Shells e elementos internos | x | | |
| GS-SW-90 | Padronização de Shells | | | x |
| GS-SW-100 | Frontend para criação/modelagem de Shells | | | x |
| GS-SW-110 | Frontend para instalação e atualização de Shells | | | x |
| GS-SW-120 | Frontend para monitoramento de Shells | | | x |

Fonte: (Elaborado pelo autor, 2023)

#### 3.1.1.1 Descrição dos Requistos Funcionais

Tabela 2 – (*RF*) Modelagem do Sistema

| **GS-SW-10** | **Modelagem do Sistema** |
|---|---|
| ***Requisito Funcional*** | Software |
| [**x**] Essencial [ ] Importante [ ] Desejável | |

**Descrição:**

- Visão geral da Arquitetura: Esquemático do Sistema, descrição do esquemático, Diagramas de caso de uso, descrição dos casos de usos;
- Modelagem Estrutural do Sistema: Diagrama de Classe e descrição das classes;
- Comportamento Dinâmico da Arquitetura: Diagrama de sequência e descrição dos fluxos de trabalho;
- Justificativa da Arquitetura: Desempenho, Segurança, Interoperabilidade, Usabilidade, Compatibilidade, Confiabilidade, Robustez, Requisitos de entrega, Padrões e Manutenibilidade;
- Diagramas de deployment, em alguns casos específicos utilizando máquinas de estado, apresentando a estratégia de uso dos Shells (Assets, Fog ).

Fonte: (Elaborado pelo autor, 2023)

Tabela 3 – (*RF*) Prototipação do Sistema (UI/UX)

| **GS-SW-20** | **Prototipação do Sistema (UI/UX)** |
|---|---|
| ***Requisito Funcional*** | Software |
| [**x**] Essencial [ ] Importante [ ] Desejável | |

**Descrição:**

- Criação de wireframes de baixa fidelidade com as funcionalidades necessárias;
- Após atestação de funcionalidades, criação das interfaces de média fidelidade;
- Comportamento Dinâmico da Arquitetura: Diagrama de sequência e descrição dos fluxos de trabalho;
- Após atestação de média fidelidade, completar as interfaces;
- Prototipação final de telas.

Fonte: (Elaborado pelo autor, 2023)

Tabela 4 – (*RF*) Criação de Shells a partir de uma base de dados

| **GS-SW-30** | **Criação de Shells a partir de uma base de dados** |
|---|---|
| ***Requisito Funcional*** | Software |
| [ ] Essencial [**x**] Importante [ ] Desejável | |

**Descrição:**

- Realizar a criação de Shells a partir de uma base de dados, sendo capaz de adicionar submódulos, propriedades, métodos, Assets e templates de Shell (Seguindo o requisito **GS-80** sobre a Biblioteca de Shells e elementos internos).

Fonte: (Elaborado pelo autor, 2023)

Tabela 5 – (*RF*) Criação de Shells do tipo arquivo

| **GS-SW-40** | **Criação de Shells do tipo arquivo** |
|---|---|
| ***Requisito Funcional*** | Software |
| [**x**] Essencial [ ] Importante [ ] Desejável | |

**Descrição:**

- Criação de Shells do tipo Arquivo: Shells do tipo arquivo não recebem métodos de comunicação, como por exemplo no formado do AASX Package Explorer.

Fonte: (Elaborado pelo autor, 2023)

Tabela 6 – (*RF*) Criação de Shells Passivos

| **GS-SW-50** | **Criação de Shells Passivos** |
|---|---|
| ***Requisito Funcional*** | Software |
| [**x**] Essencial [ ] Importante [ ] Desejável | |

**Descrição:**

- Criação de Shells do tipo Passivo: Shells que recebem métodos de comunicação porém não são atores na comunicação e respondem solicitações. Tais Shells deverão receber a associação de um método *OPC UA* com uma função na linguagem de programação em qual foi concebido para execução da lógica necessária para a sua execução.

Fonte: (Elaborado pelo autor, 2023)

Tabela 7 – (*RF*) Criação de Shells Ativos

| **GS-SW-60** | **Criação de Shells Ativos** |
|---|---|
| ***Requisito Funcional*** | Software |
| [**x**] Essencial [ ] Importante [ ] Desejável | |

**Descrição:**

- Criação de Shells do tipo Ativo: Shells que recebem métodos de comunicação, são atores nas requisições e também respondem solicitações. Tais Shells deverão receber a associação de um método OPC UA com uma função na linguagem de programação em qual foi concebido para execução da lógica necessária para a sua execução.

Fonte: (Elaborado pelo autor, 2023)

Tabela 8 – (*RF*) Edição de Shells template

| **GS-SW-70** | **Edição de Shells template** |
|---|---|
| ***Requisito Funcional*** | Software |
| [**x**] Essencial [ ] Importante [ ] Desejável | |

**Descrição:**

- Realizar edição de templates, acrescentando ou retirando submodelos, propriedades e métodos já existentes.

Fonte: (Elaborado pelo autor, 2023)

Tabela 9 – (*RF*) Biblioteca de Shells e elementos internos

| **GS-SW-80** | **Biblioteca de Shells e elementos internos** |
|---|---|
| ***Requisito Funcional*** | Software |
| [**x**] Essencial [ ] Importante [ ] Desejável | |

**Descrição:**

- Capacidade de reutilizar ou criar insumos para a construção de novos Shells a partir de uma biblioteca local da empresa;
- Desenvolvimento de uma biblioteca local com submodelos, propriedades, atributos de propriedades, Assets e templates de Shells;
- Para padronizar as propriedades, implementaremos uma funcionalidade que avaliará se o Shell atende aos padrões definidos pela empresa.

Fonte: (Elaborado pelo autor, 2023)

Tabela 10 – (*RF*) Padronização de Shells

| **GS-SW-90** | **Padronização de Shells** |
|---|---|
| ***Requisito Funcional*** | Software |
| [**x**] Essencial [ ] Importante [ ] Desejável | |

**Descrição:**

- Validar a estrutura interna do Shell (submódulos e propriedades) e sua forma de comunicação com a rede, mantendo a consistência de módulos, propriedades e Assets;
- Trabalho de verificação contínua, avaliando a consistência do Shell na rede, atendendo os requisitos mínimos de um Shell funcional;
- Realizar a verificação da interface dos métodos de comunicação do Shell, avaliando a padronização de sua estrutura. Verificação final após o parser, antes de o Shell se tornar visível na rede.

Fonte: (Elaborado pelo autor, 2023)

Tabela 11 – (*RF*) Frontend para criação/manipulação de Shells

| **GS-SW-100** | **Frontend para criação/manipulação de Shells** |
|---|---|
| ***Requisito Funcional*** | Software |
| [ ] Essencial [ ] Importante [**x**] Desejável | |

**Descrição:**

- Frontend representando o sub requisito de criação, edição e importação de Shells;
- Desenvolver um frontend seguindo padrões do AASX Package Explorer.

Fonte: (Elaborado pelo autor, 2023)

Tabela 12 – (*RF*) Frontend para instalação e atualização de Shell

| **GS-SW-110** | **Frontend para instalação e atualização de Shell** |
|---|---|
| ***Requisito Funcional*** | Software |
| [ ] Essencial [**x**] Importante [] Desejável | |

**Descrição:**

- Desenvolver um frontend para facilitar o deploy e update de Shells modelados no sistema baseado do no requisito GS-WS-100
- Front-end representando os passos para a instalação e atualização de Shells.

Fonte: (Elaborado pelo autor, 2023)

Tabela 13 – (*RF*) Frontend para monitoramento de Shells

| **GS-SW-120** | **Frontend para monitoramento de Shells** |
|---|---|
| ***Requisito Funcional*** | Software |
| [**x**] Essencial [ ] Importante [ ] Desejável | |

**Descrição:**

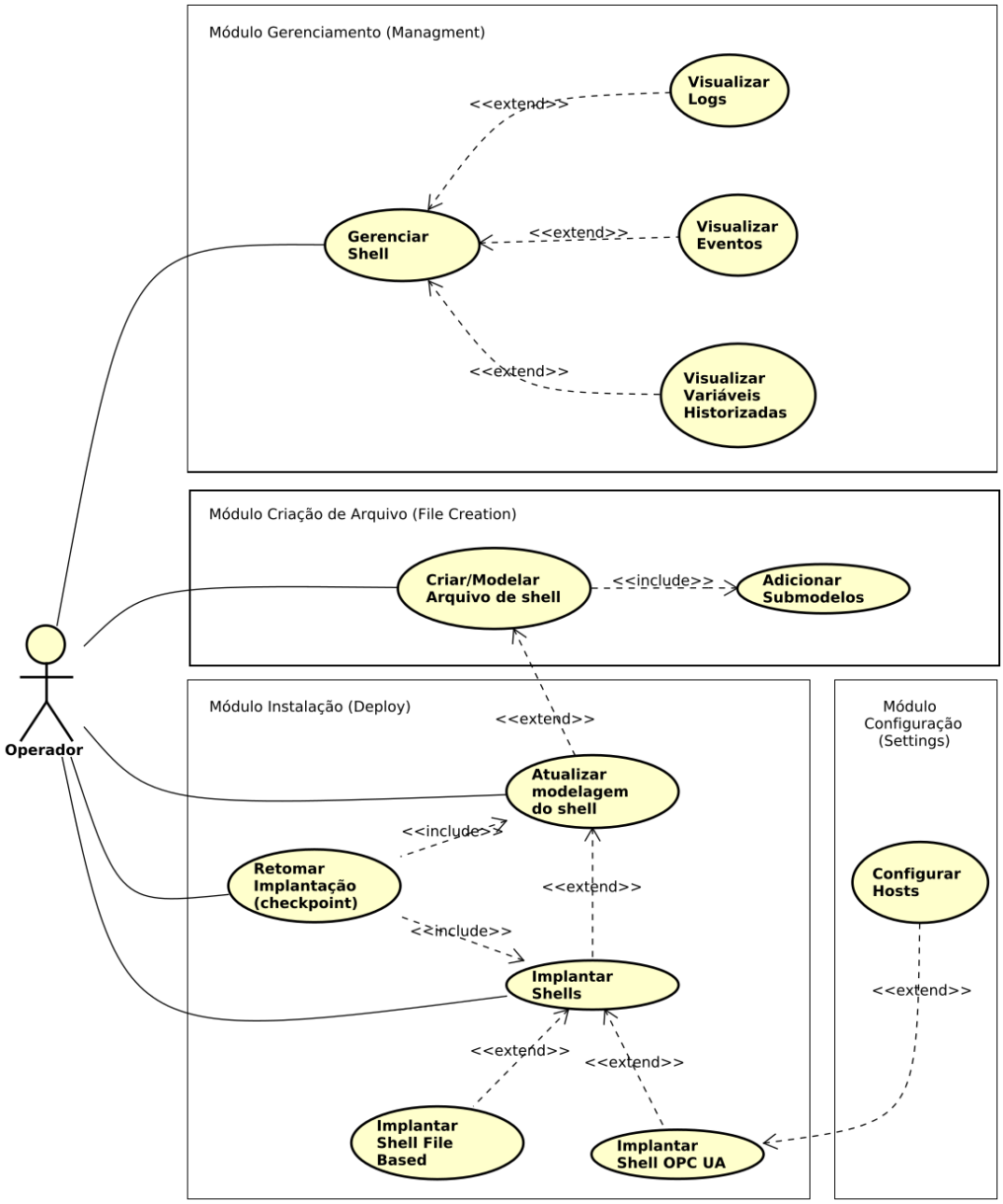
- Desenvolver um front-end para o gerenciamento dos Shells instalado pelo sistema baseado no requisito GS-110.
- Apresentar gráfico com os eventos recorrentes no Shells.
- Possibilitar a visualização de logs dos Shells instalados pelo sistema

Fonte: (Elaborado pelo autor, 2023)

## 3.2 Casos de Uso

A Figura 10 representa o diagrama de caso de uso do sistema de gerenciamento de Shell que são documentos textuais de especificação de requisitos. Além disso, contém todos os casos de uso sendo categorizado com base em qual módulo pertence.

Figura 10 – Diagrama de caso de uso




Fonte: (Elaborado pelo autor, 2023)

### 3.2.1 Descrição de Casos de Uso

Nesta secção estão detalhados todos os casos de uso relacionados a sistema.

Tabela 14 – (Caso de Uso) Criar/Modelar Shell

| **UC001 – Criar/Modelar Shell** | |
|---|---|
| **Ator:** | Operador |
| **Descrição:** | O Operador pode criar Shells e salvá-los na biblioteca, assim como seus submodelos. |
| **Fluxo Normal:** | 1. O operador seleciona o módulo de criação de arquivo (File Creation).<br>2. O sistema mostra o modelo de criação de arquivo de um Shell.<br>3. O operador insere um Asset administrations Shell (AAS).<br>4. Operador insere um Asset e o associa ao AAS.<br>5. Operador salva o arquivo criado.<br>6. O sistema disponibiliza o arquivo do Shell na biblioteca. |
| **Extensões:** | 5a. Se a modelagem for necessária, cancelar a criação. |

Fonte: (Elaborado pelo autor, 2023)

Tabela 15 – (Caso de Uso) Adicionar Submodelos

| **UC002 – Adicionar Submodelos** | |
|---|---|
| **Ator:** | Operador |
| **Descrição:** | O Operador pode adicionar submodelos ao *AAS* ou o sistema insere a partir da biblioteca. |
| **Fluxo Normal:** | 1. (Caso de Uso) Criar/Modelar Shell.<br>2. O operador selecionar submodelos da biblioteca.<br>3. O sistema insere automaticamente os submodelos ao Shell. |
| **Extensões:** | 2a. Operador pode criar submodelos manualmente conforme a necessidade. |

Fonte: (Elaborado pelo autor, 2023)

Tabela 16 – (Caso de Uso) Implantar Shell *OPC UA*

| **UC003 – Implantar Shell *OPC UA*** | |
|---|---|
| **Ator:** | Operador |
| **Descrição:** | O Operador pode realizar a instalação de um Shell que possui comunicação usando o padrão *OPC UA*. |
| **Fluxo Normal:** | 1.Operador seleciona o módulo de instalação (Deploy)<br>e a opção arquivo externo (External File).<br>2. Operador seleciona a opção de Shell com OPC UA.<br>3. Operador seleciona que o Shell não é baseado em um<br>template.<br>4. O sistema verifica a estrutura do Shell através dos passos<br>de instalação.<br>5. O operador seleciona o servidor onde será instalado<br>a instância do Shell.<br>6. O sistema emite uma confirmação de sucesso na instalação.<br>7. O operador pode finalizar e realizar o download do QRcode<br>de acesso ao Shell. |
| **Extensões:** | 3a. Se o Shell for baseado em um template, selecionar<br>a opção de template.<br>5a. Se o Shell for do tipo arquivo não há seleção de servidor.<br>6a. Se a instalação falhar, emitir uma alerta de erro<br>para o usuário.<br>7a. Se visualizar o Shell instalado for selecionado, redirecionar<br>para homologação do módulo Deploy.<br>7b. Se for o operador apenas finalizar, redirecionar para a home<br>do módulo Deploy |

Fonte: (Elaborado pelo autor, 2023)

Tabela 17 – (Caso de Uso) Implantar Shell File Based

| **UC004 – Implantar Shell File Based** | |
|---|---|
| **Ator:** | Operador |
| **Descrição:** | O Operador pode realizar a implantação de um Shell que não possui comunicação, apenas uma representação digital. |
| **Fluxo Normal:** | 1. Operador seleciona o módulo de instalação (Deploy) e seleciona opção arquivo externo.<br>2. Operador seleciona a opção de Shell do tipo arquivo (file based).<br>3. Operador seleciona a possibilidade o Shell ser baseado em um template.<br>4. O sistema verifica a estrutura do Shell através dos passos de atualização.<br>5. O sistema emite uma confirmação .<br>6. O Operador pode visualizar o Shell atualizado na opção de homologação do módulo Deploy. |
| **Extensões:** | Não há. |

Fonte: (Elaborado pelo autor, 2023)

Tabela 18 – (Caso de Uso) Atualizar modelagem do Shell

| **UC005 – Atualizar modelagem do Shell** | |
|---|---|
| **Ator:** | Operador |
| **Descrição:** | O Operador pode realizar a atualização da modelagem, bem como servidor do Shell instalado. |
| **Fluxo Normal:** | 1. O operador seleciona o módulo de instalação (Deploy).<br>2. O operador pode selecionar opção *"Shells under homologation"*.<br>3. O operador seleciona a opção atualizar (update) do respectivo Shell.<br>4. O operador confirma a ação de atualizar o Shell.<br>5. O sistema verifica a estrutura do novo Shell através dos passos de instalação.<br>6. O sistema emite uma confirmação de sucesso na atualização.<br>7. O operador pode realizar o download do QRcode de acesso ao Shell. |
| **Extensões:** | 3a. Se a opção "Delete" for selecionada, deletar instância do Shell.<br>3b. Se exclusão não é necessária, cancelar ação.<br>4a. Se atualização não é necessária, cancelar ação.<br>6a. Se a atualização falhar, emitir uma alerta de erro para o usuário.<br>7a. Se visualizar o Shell instalado for selecionado, redirecionar para homologação do módulo Deploy.<br>7b. Se for o operador apenas finalizar, redirecionar para a home do módulo Deploy. |

Fonte: (Elaborado pelo autor, 2023)

Tabela 19 – (Caso de Uso) Retomar implantação (checkpoint)

| **UC006 – Retomar implantação (checkpoint)** | |
|---|---|
| **Ator:** | Operador |
| **Descrição:** | O Operador pode retomar o passo onde<br>a instalação/atualização<br>foi pausada ou interrompida. |
| **Fluxo Normal:** | 1. O operador seleciona o módulo de instalação (Deploy).<br>2. O operador seleciona a opção “checkpoint”.<br>3. O sistema retoma o último passo do processo de<br>(Caso de Uso) Implantar Shell File Based |
| **Extensões:** | 3a. Se checkpoint for de atualização, redirecionar para<br>(Caso de Uso) Atualizar modelagem do Shell |

Fonte: (Elaborado pelo autor, 2023)

Tabela 20 – (Caso de Uso) Configurar hosts

| **UC007 – Configurar hosts** | |
|---|---|
| **Ator:** | Operador |
| **Descrição:** | O Operador pode retomar o passo onde a instalação/atualização foi pausada ou interrompida. |
| **Fluxo Normal:** | 1. O operador seleciona o módulo de configuração (settings).<br>2. O operador seleciona a opção “Default Host”.<br>3. O sistema exibe a lista com hosts disponíveis na rede.<br>4. O operador seleciona o host e insere credenciais de acesso ssh usuário e senha.<br>6. O sistema verifica credenciais como tentativa de acesso usando ssh.<br>7. O sistema emite uma confirmação de sucesso para a configuração do host.<br>8. O sistema exibe lista com os hosts atualizados. |
| **Extensões:** | 5a. Se o servidor for de comissionamento, selecionar a opção "Commissioning Default”.<br>7a. Se falhar na configuração do host, emitir alerta de erro. |

Fonte: (Elaborado pelo autor, 2023)

Tabela 21 – (Caso de Uso) Visualizar Logs

| **UC008 – Visualizar Logs** | |
|---|---|
| **Ator:** | Operador |
| **Descrição:** | O Operador pode visualizar os logs dos Shells implantados pelo sistema. |
| **Fluxo Normal:** | 1. O operador seleciona o módulo de gerenciamento (Management).<br>2. O operador seleciona a listagem de Shells implantados.<br>3. O operador seleciona opção de visualização de logs do Shell<br>4. O sistema exibe a caixa de logs do Shell selecionado, atualizando a cada 5 segundos. |
| **Extensões:** | Não há |

Fonte: (Elaborado pelo autor, 2023)

Tabela 22 – (Caso de Uso) Visualizar variáveis historizadas

| **UC009 – Visualizar variáveis historizadas** | |
|---|---|
| **Ator:** | Operador |
| **Descrição:** | O Operador pode visualizar as variáveis historizadas dos Shells implantados pelo sistema. |
| **Fluxo Normal:** | 1. O operador seleciona o módulo de gerenciamento (Management).<br>2. O operador seleciona a listagem de Shells implantados.<br>3. O operador seleciona a opção de histórico de variáveis do Shell.<br>4. O sistema exibe a modal de seleção de variáveis a serem visualizadas.<br>5. O operador informa o intervalo de dados e as variáveis a serem visualizadas.<br>6. O sistema exibe modal com os gráficos das variáveis selecionadas. |
| **Extensões:** | 6a. Se o intervalo de dados for cíclico, exibir dados em tempo real. |

Fonte: (Elaborado pelo autor, 2023)

Tabela 23 – (Caso de Uso) Visualizar Eventos

| **UC010 – Visualizar Eventos** | |
|---|---|
| **Ator:** | Operador |
| **Descrição:** | O Operador pode visualizar os eventos relacionados aos Shells implantados pelo sistema. |
| **Fluxo Normal:** | 1. O operador seleciona o módulo de gerenciamento (Management).<br>2. O operador seleciona a listagem de Shells implantados.<br>3. O operador seleciona a opção de eventos.<br>4. O sistema exibe listagem de eventos monitorados, ordenados por data. |
| **Extensões:** | Não há. |

Fonte: (Elaborado pelo autor, 2023)

## 3.3 Arquitetura

A arquitetura do software do gerenciador de Shells está estruturada em componentes e em interações. Estas escolhas de design estão relacionadas com os resquisitos funcionais e não-funcionais previamente definidos e outras restrições, mas estas escolhas, por sua vez, acrescentam mais restrições aos requisitos e decisões futuras de concepção a um nível inferior. A arquitectura do gestor da Shell está preocupada com os seguintes aspectos:

- A estrutura do modelo: padrões organizacionais, por exemplo, Camadas.
- Os elementos essenciais: casos de uso crítico, classes principais, mecanismos comuns, etc., ao contrário de todos os elementos presentes no modelo.
- Alguns dos principais cenários que mostram os principais fluxos de controle de todo o sistema.
- Serviços para captar a capacidade de divisão em módulos, características opcionais, e aspectos dos produtos de linha.

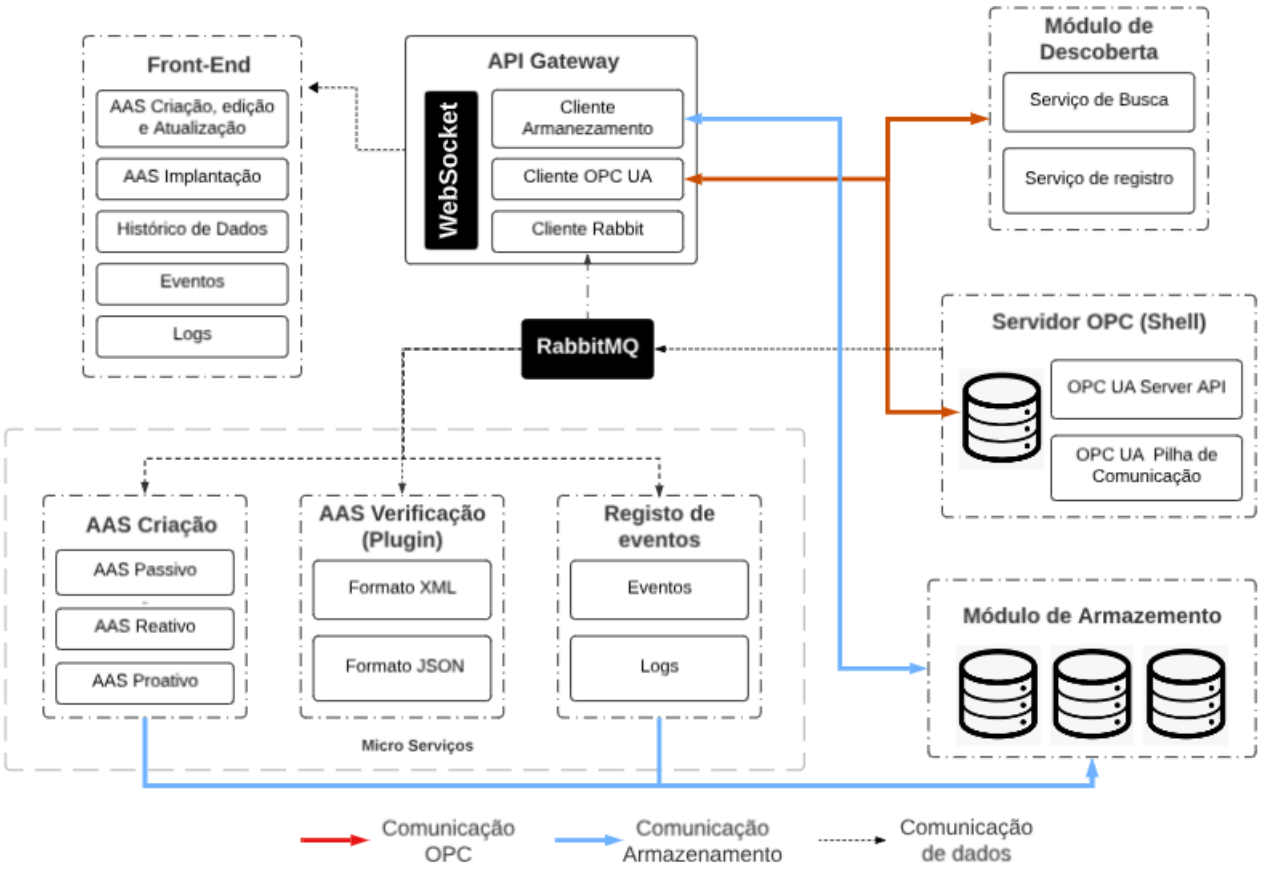
A representação do sistema será feita através de diagramas esquemáticos, diagrama de caso de uso, componente Diagrama que engloba o sistema de gerenciamento de Shell.

### 3.3.1 Diagrama Arquitetural do Sistema

A figura 11 contém a diagramação da arquitetura do gerenciamento de Shell. O diagrama sistemático do gerenciador de Shell consiste em partes que facilitam o processo de desenvolvimento. Cada módulo foi concebido de acordo com as recomendações das fundações RAMI 4.0 e *OPC UA*. O sistema foi concebido com base nos requistos descritos na secção Requisitos do Sistema.

O gerenciador de Shells é baseado em duas plataformas bem conhecidas no contexto da *Indústria 4.0*.

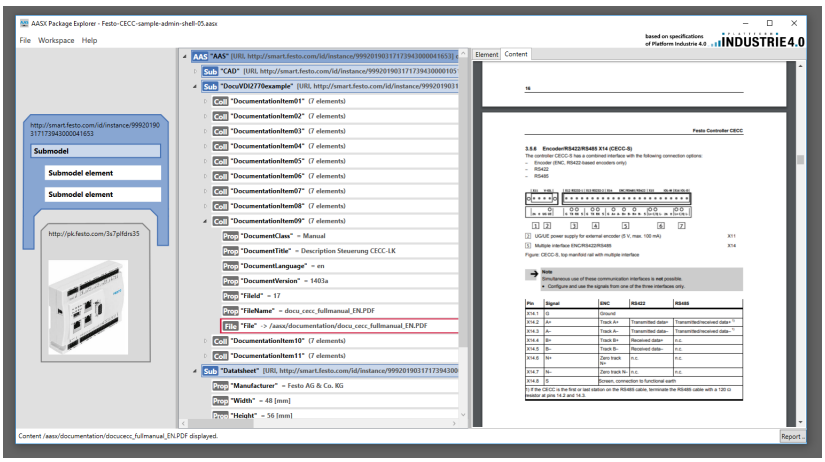
Figura 11 – Diagrama de Arquitetura




Fonte: (Elaborado pelo autor, 2023)

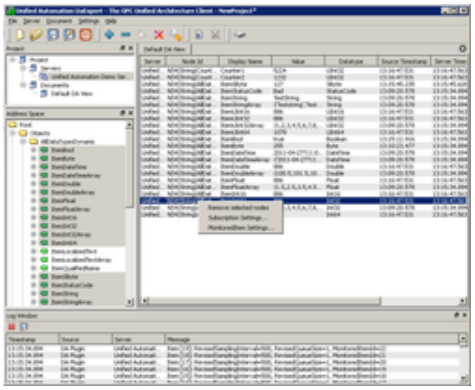
- **AASX Package Explorer:** Visualizador/editor baseado em C# para *Asset Administration Shell*s representado na figura 12.

Figura 12 – Software AASX Package Explorer



Fonte: (IDTA, 2023)

Figura 13 – Software UA Expert



Fonte: (Unified Automation, 2023)

#### 3.3.1.1 Front-End

O front-end é referido como o lado cliente da aplicação. Ele inclui tudo o que os usuários experimentam diretamente: cores e estilos de texto, imagens, gráficos e tabelas, botões, cores e um menu de navegação. foram utilizadas linguagens HTML, CSS e JavaScript

para o desenvolvimento do Front-end. Responsividade e desempenho são os dois principais objetivos.

O interface do sistema de gerenciamento de *AAS* é fundamental para proporcionar aos usuários uma visão clara e controlada dos ativos digitais, facilitando a administração e a tomada de decisões estratégicas. E foi projetado para atender às necessidades específicas do ambiente industrial em que foi implementado.

#### 3.3.1.2 API Gateway

Uma API Gateway é um estrutura de software que atua como um intermediário entre diferentes serviços e clientes em uma arquitetura de microsserviços. Este serviço roteia as requisições dos clientes para os serviços apropriados. Isso é especialmente útil em ambientes com múltiplos serviços que precisam ser acessados de forma transparente.

O gateway do sistema de gerenciamento de Shells possui algumas características para englobar as funcionaliades:

- **Websocket:** é um protocolo independente baseado em TCP que permite que você evite parte da sobrecarga, e possivelmente maior latência, de HTTP. Para estabelecer uma conexão WebSocket, o cliente envia uma solicitação HTTP normal que usa semântica de atualização do HTTP para alterar o protocolo. O servidor pode concluir o handshake. A conexão WebSocket permanece aberta, e o cliente ou servidor pode enviar dados quadros entre si sem a necessidade de estabelecer novas conexões a cada vez (Amazon Web Services, 2023). Este protocolo está sendo utilizada para a monitoramento de eventos, logs e status dos gêmeos digitais.

- **Cliente OPC UA:** envia pacotes de mensagens para dispositivos de servidor e recebe respostas de seus dispositivos de servidor (JOHN S RINALDI, 2018). Está sendo utilizado para abrir comunicações seguras e confiáveis com os *Asset Administration Shell*s. *OPC UA* é uma arquitetura orientada a serviços independente de plataforma que integra todas as funcionalidades das especificações individuais do OPC Classic em uma estrutura extensível (OPC Foundation, 2023).

- **Cliente Rabbit:** Aceita mensagens de editores, as direciona e, se houvesse filas para onde ir, as armazena para consumo ou imediatamente entrega aos consumidores, se houver (RabbitMQ, 2023).
- **Cliente Armazenamento:** interage com o módulo de armazenamento. Esse cliente é responsável por realizar operações como armazenar, recuperar e gerenciar dados dentro do sistema de armazenamento.

#### 3.3.1.3 Verificação AAS (Compliance)

O serviço de verificação de compliance refere-se à validação e garantia de que os *AAS*s e os ativos digitais associados estão em conformidade com os padrões, regulamentações e diretrizes estabelecidos pela empresa que os estão implantando. Este serviço é importante para garantir a eficácia, segurança e interoperabilidade dos sistemas e processos na *Indústria 4.0*.

Além é responsável pela validação dos formatos aceitos pelo sistema de gerenciamento de Shell.

#### 3.3.1.4 Criação AAS (Creation)

Este serviço envolve vários passos para definir, modelar e implementar uma representação digital de um ativo físico de forma padronizada e interoperável.

Inicialmente é necessário quais aspectos do ativo físico é desejado representar digitalmente. É identificado as propriedades, características chave do ativo físico que precisam ser representados no *AAS*, incluindo propriedades tanto estáticas quanto dinâmicas.

Submodelos são definidos para diferentes aspectos do ativo, como configuração, comportamento e estado. Além disso, é necessário especificar cada propriedade, método e evento, tipos de dados, unidades e quaisquer restrições ou dependências, junto com suas descrições de conceitos ao modelo para fornecer contexto e significado adicionais aos dados, facilitando a interpretação.

Após isso, são eleitos os protocolos de comunicação e interfaces que serão usados para interagir com o *AAS*. O que incli definir como outros sistemas podem acessar e se comunicar com o *AAS*. Uma vez que a modelagem do *AAS* estiver completa e validada, ela pode implementada no ambiente industrial onde será usada.

#### 3.3.1.5 Registro de Eventos (Event Logging)

O registro de eventos, também conhecido como Event Logging, consiste em manter um registro detalhado e cronológico de eventos significativos que ocorreram dentro do sistema de gerenciamento, dentro dos gêmeos digitais implantados. Esses eventos incluem ações, erros, alertas, notificações, alterações de estado, entre outros.

O registro de eventos fornece um histórico completo das atividades que ocorreram dentro do sistema ou do ambiente, permitindo uma análise retrospectiva, o que facilita a identificação e a resolução de problemas, uma vez que oferece informações detalhadas sobre as circunstâncias que levaram a um evento específico.

Com isso, proporcionamdo uma base sólida para monitoramento, análise e tomada de decisões, promovendo uma operação mais eficiente, segura e confiável dos ativos monitorados.

#### 3.3.1.6 Serviço de Descoberta (Discovery)

Este serviço implementa *Local Discovery Service* (*LDS*), que refere-se a um serviço de descoberta local associado a um *AAS* (*AAS*) em um contexto de *Indústria 4.0*.

O *LDS* desempenha um papel importante na identificação e localização de serviços e informações específicas relacionadas a ativos digitais representados pelo *AAS* em um ambiente industrial. Ele facilita a comunicação e a interoperabilidade entre sistemas, permitindo que componentes na rede industrial descubram e interajam com os ativos digitais.

#### 3.3.1.7 Serviço de Armazenamento (Storage)

Este serviço lida com o armazenamento e a gestão de dados relacionados aos ativos digitais representados pelo *AAS*. Esta capacidade é fundamental para manter e acessar informações relevantes sobre os ativos ao longo do tempo.

O serviço é responsável por armazenar dados pertinentes aos ativos digitais, como informações de configuração, histórico de operações, estado atual e outros detalhes relevantes. Permite a criação de um histórico de variáveis de ativos, o que é essencial para análises retrospectivas, manutenção preditiva e rastreamento de tendências.

Com issdo, é capaz de lidar com uma variedade de tipos de dados, como texto, numérico, imagens, dependendo das necessidades dos ativos representados. Além disso, pode suportar a gestão de dados em tempo real, permitindo a captura e o armazenamento de dados instantaneamente à medida que são gerados pelos ativos.

# 4

# RESULTADOS

Nesta seção, apresentaremos os resultados obtidos a partir da análise meticulosa dos dados coletados durante o curso deste projeto. Os dados foram organizados e interpretados com o intuito de responder à questão central deste estudo: Gerenciamento de Shell no contexto da indústria 4.0. Com o objetivo de oferecer uma visão abrangente e coerente do desenvolvimento, os resultados serão apresentados de acordo com os principais resquisitos levantados.

## 4.1 Criação e modelagem de Shells

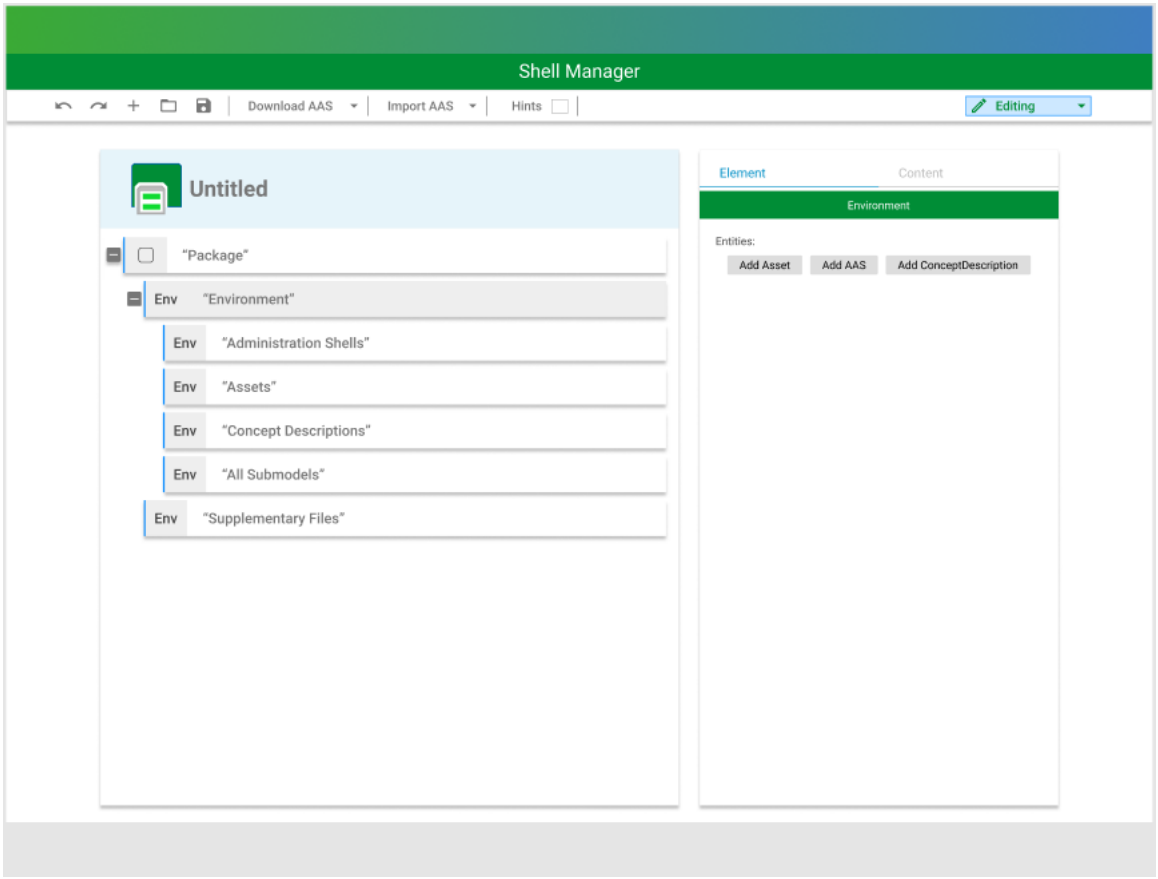
Ao entrar na página um modelo base em branco é carregado na parte esquerda, possibilitando o início da modelagem, no lado direito são exibidas os atributos e ações que o operador pode executar durante a criação ou atualização do Shell. A figura 14 exibe a página do módulo de criação de Shells.

O modelo base é constituído pelas seguintes secções:

- **Package (Pacote):** é um contêiner que agrupa diferentes partes de um *AAS*, permitindo uma organização mais eficiente e modular das informações.
  - **Environment (Ambiente):** descreve o contexto e o ambiente em que os ativos são utilizados. Inclui informações como a configuração física, localização, estado operacional, e outros dados relacionados ao ambiente onde os ativos estão inseridos.

    - **Asset Administration (Administração de Ativos):** dedicada à administração e gestão dos ativos. Ela contém informações sobre a identificação, configuração, manutenção e vida útil dos ativos. Também pode incluir detalhes sobre a propriedade, condição, histórico de manutenção, e outras informações relacionadas à administração do ativo.
    - **Assets (Ativos):** contém informações específicas sobre os ativos individuais. Ela pode incluir detalhes como a identificação única do ativo, características técnicas, especificações de desempenho, histórico de uso, e outras informações relevantes para cada ativo em particular.
    - **Concept Descriptions (Descrições de Conceito):** fornece descrições detalhadas dos conceitos e características dos ativos e suas relações. Pode incluir informações sobre as funcionalidades, interfaces, propriedades e comportamentos dos ativos.
    - **All Submodels (Todos os Submodelos):** Esta seção pode conter todos os submodelos associados ao *AAS*, permitindo uma visão completa e detalhada de todos os ativos e suas relações no contexto do *AAS*.
  - **Supplementary Files (Arquivos Suplementares):** pode conter documentos, imagens, manuais ou quaisquer outros arquivos suplementares que fornecem informações adicionais sobre os ativos ou o ambiente em que estão inseridos.

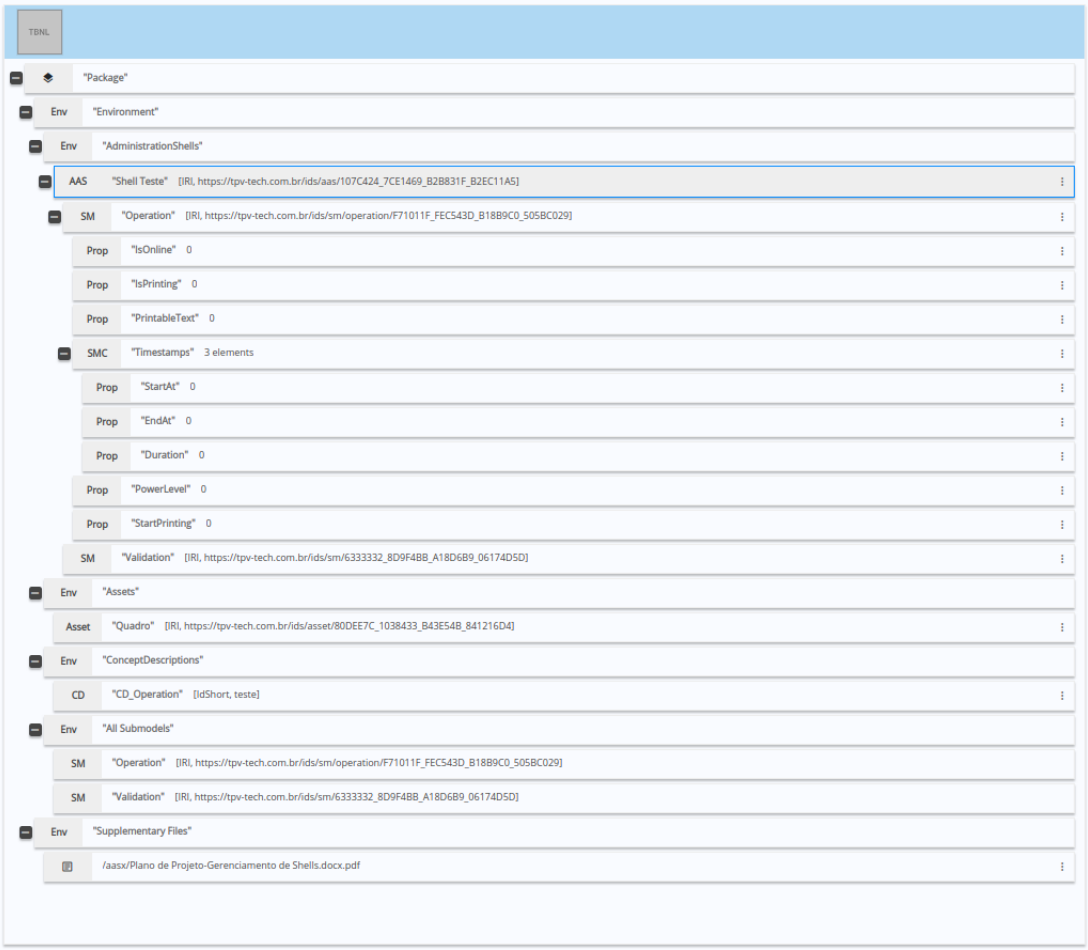
Figura 14 – Página inicial para criação de Shells




Fonte: (Elaborado pelo autor, 2023)

De forma a exemplificar um *Asset Administration Shell* modelado pelo sistema, é demonstrado na figura 15 um *AAS* com seu respectivo Asset (ativo) associado. Além disso, esse Shell possui um submodelo de operação com suas propriedades referenciadas de forma que possam ser acessíveis atráves de sua comunicação após ser implantado ou atualizado pelo módulo de implatanção do sistema proposto neste documento.

Figura 15 – Shell modelado no módulo de criação

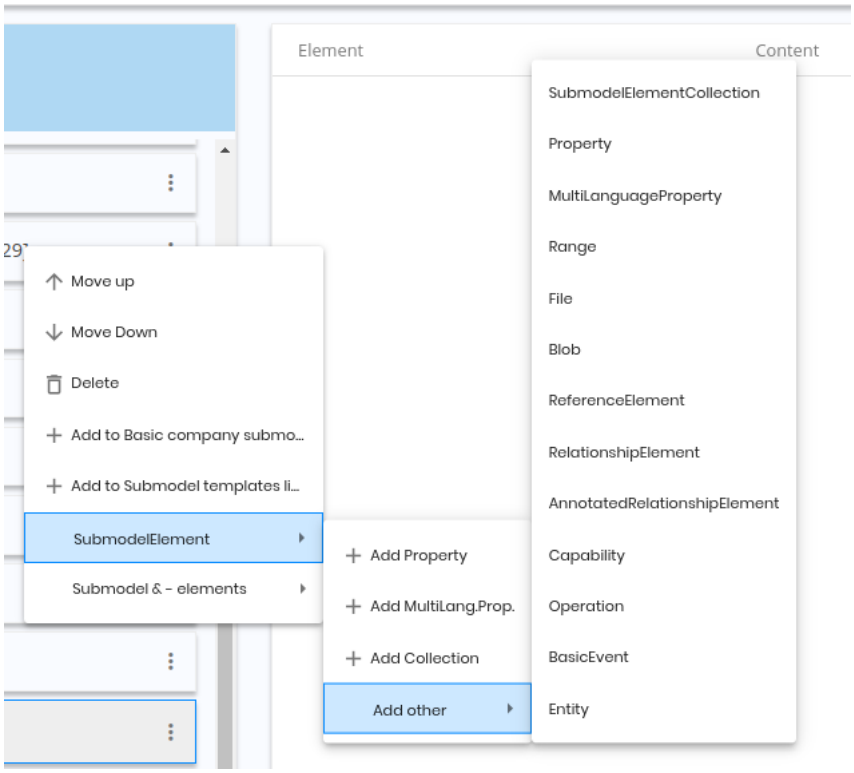



Fonte: (Elaborado pelo autor, 2023)

## 4.2 Metamodelos

Esta subseção descreve os metamodelos que foram implementados no módulo de criação e modelagem de Shells como demontrado na figura 16.

Figura 16 – Menu de metamodelos

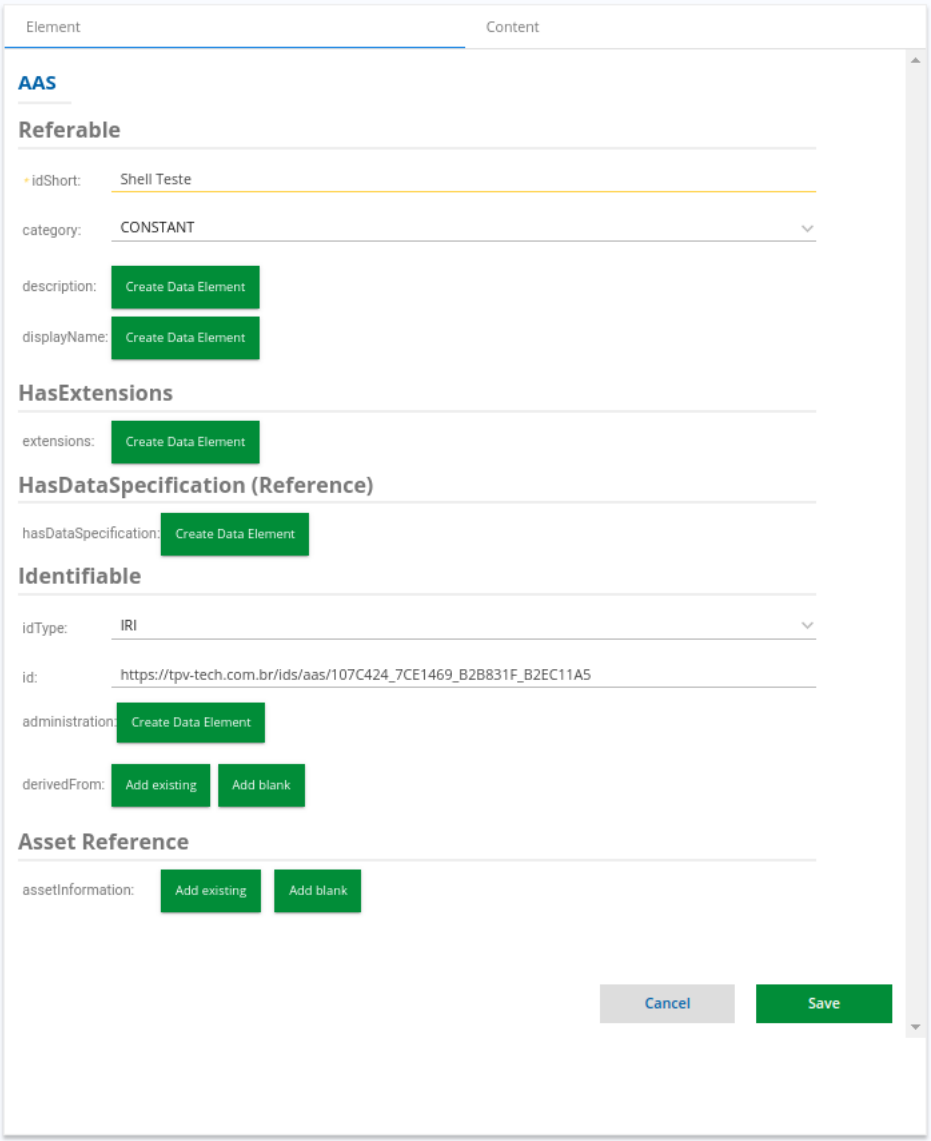



Fonte: (Elaborado pelo autor, 2023)

### 4.2.1 Asset Administration Shell

Na figura 17 é demonstrado a implementação propriamente dita das propriedades possíveis de um *Asset Administration Shell*, é identificável de forma exclusiva, pois herda de Identifiable. O atributo derivedFrom é usado para estabelecer um relacionamento entre dois *AAS*s que são derivados um do outro (Plattform Industrie 4.0, 2023).

Figura 17 – Metamodelo Asset

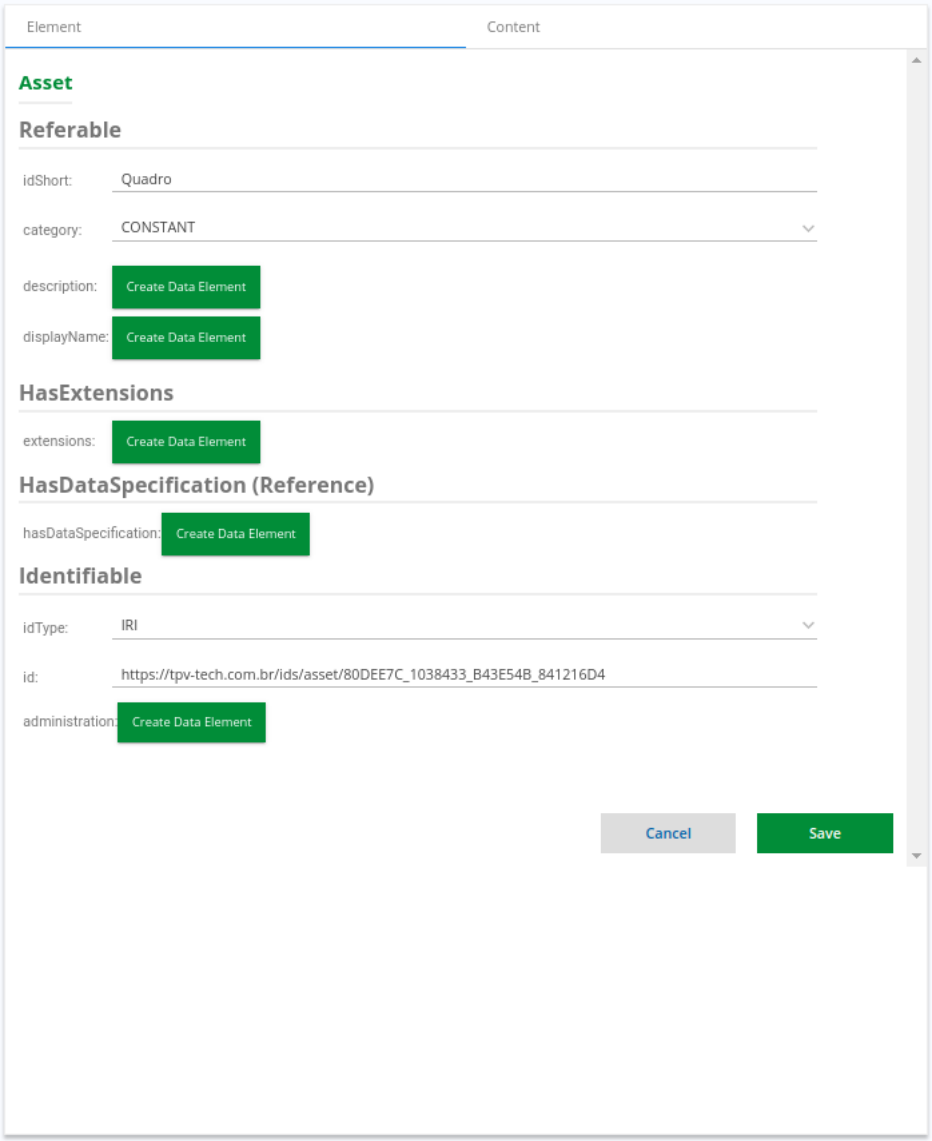



Fonte: (Elaborado pelo autor, 2023)

### 4.2.2 Asset

No Asset, são definidos os metadados de identificação do ativo que é representado por um *AAS* definido. O ativo pode representar um tipo de ativo ou uma instância de ativo. Além disso, o ativo tem um identificador globalmente exclusivo e, se necessário, identificadores adicionais específicos do domínio (proprietários) específicos do domínio (Plattform Industrie 4.0, 2023). Na figura 18 é representado a classe Asset ou Asset information.

Figura 18 – Metamodelo Asset

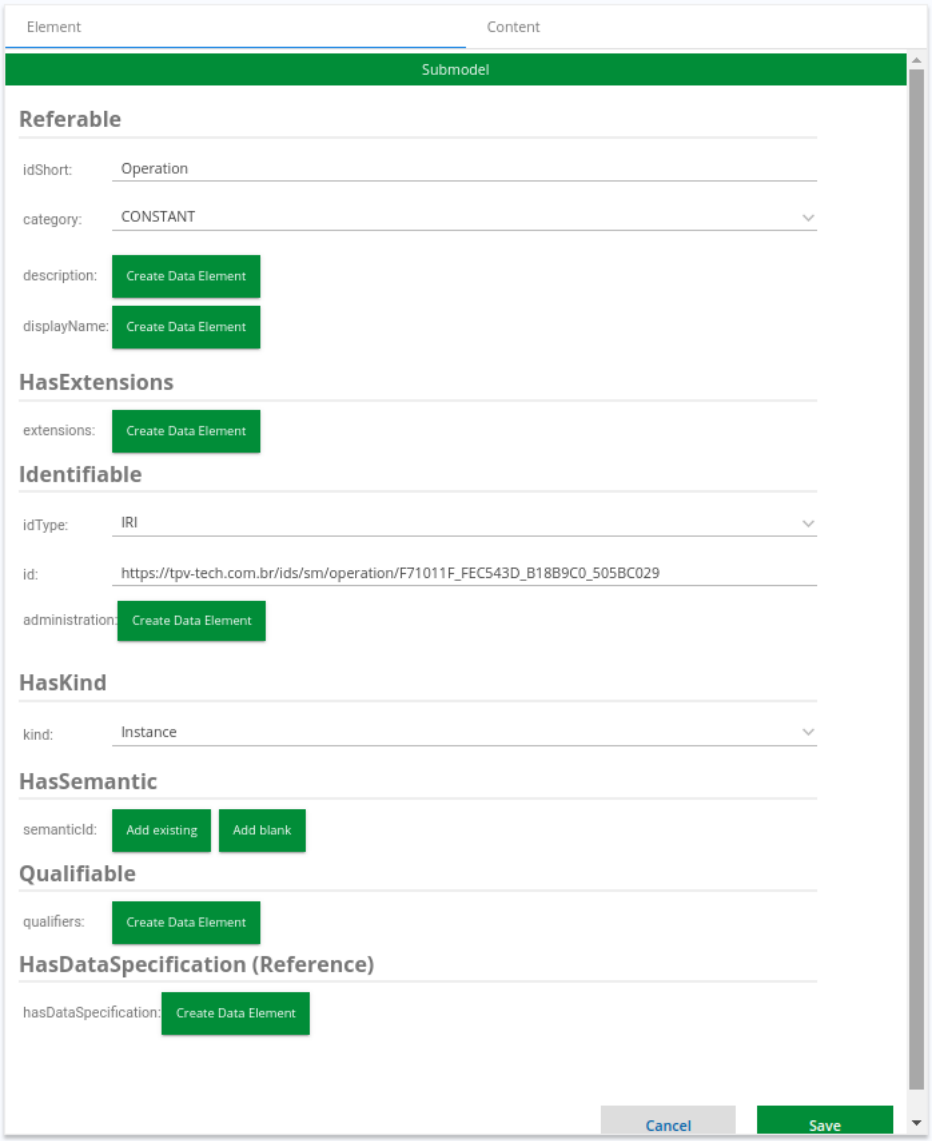



Fonte: (Elaborado pelo autor, 2023)

### 4.2.3 Submodelo (Submodel)

Um submodelo é usado para estruturar a representação digital e a funcionalidade técnica de um Shell em partes distinguíveis. Na figura 19 é representado a classe submodelo, que define um aspecto específico do ativo representado pelo *AAS*. Cada submodelo refere-se a um domínio ou assunto bem definido. Ele podem ser padronizados e, portanto, tornar-se submodelos templates, (Plattform Industrie 4.0, 2023).

Figura 19 – Metamodelo Submodel




Fonte: (Elaborado pelo autor, 2023)

#### 4.2.3.1 Tipos de Elementos do Submodelo

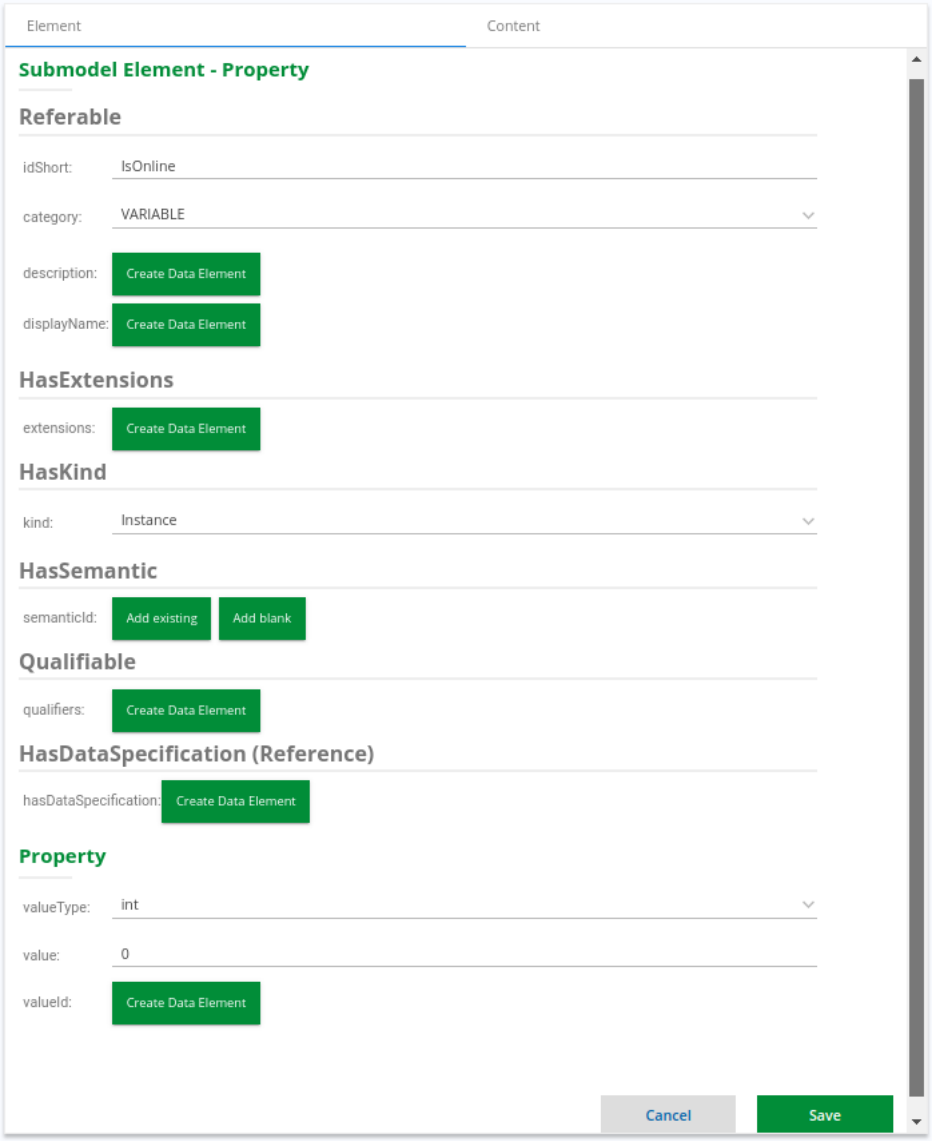
Os elementos do submodelo incluem propriedades de dados, bem como operações, eventos e outros elementos necessários para descrever um modelo para um ativo. Abaixo é demonstrado os tipos de elementos do submodelo que foram implementados com base no documento de especificação e detalhes do *Asset Administration Shell*.

##### 4.2.3.1.1 Propriedade (Property)

O submodelo propriedade, que é demonstrado na figura 20, é um elemento de dados que tem um único valor. Se ambos, value e

valueId, estiverem presentes, o valor de value precisa ser idêntico ao valor codificado referenciado em valueId (Plattform Industrie 4.0, 2023).

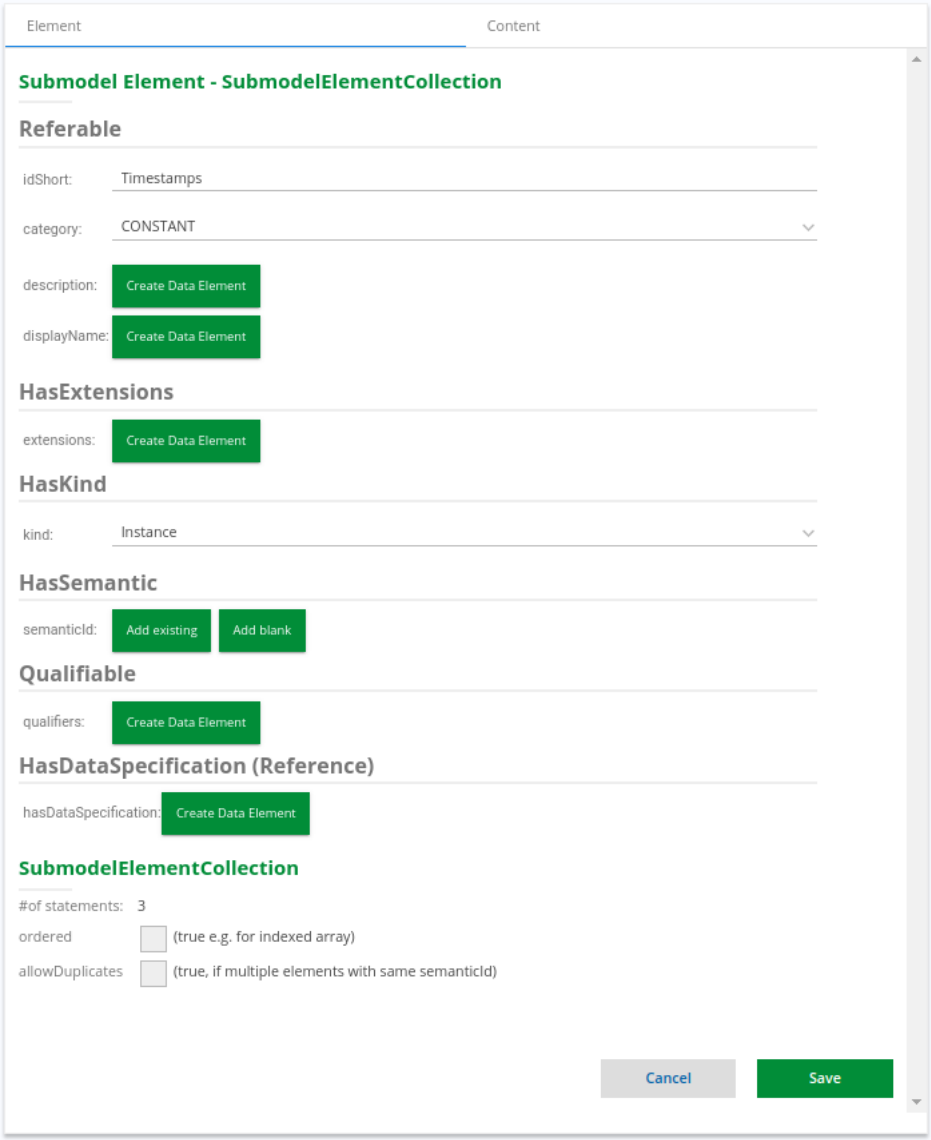
Figura 20 – Submodelo Propriedade




Fonte: (Elaborado pelo autor, 2023)

4.2.3.1.2 Coleção (Submodel Element Collection) Os submodelos de coleção, representado pela figura 21, são usados para entidades com um conjunto fixo de propriedades, com nomes exclusivos dentro da estrutura. Cada propriedade dentro da coleção deve ter uma semântica claramente definida. Uma propriedade de uma estrutura pode ser qualquer submodelo elemento que contenha um valor (Plattform Industrie 4.0, 2023).

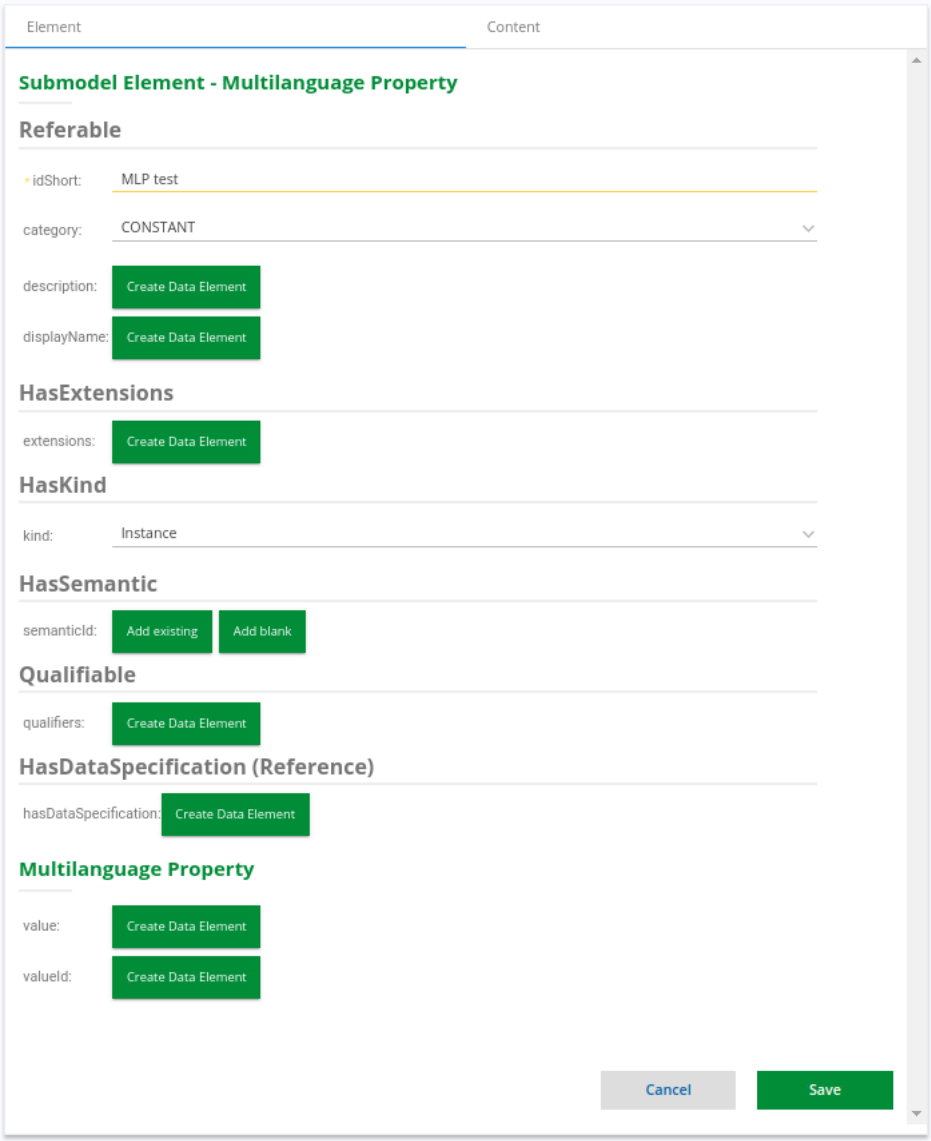
Figura 21 – Submodelo de Coleção




Fonte: (Elaborado pelo autor, 2023)

4.2.3.1.3 Multi Linguagem (Multi Language Property) O submodelo multi linguagem, representado pela figura 22, é um elemento de dados que tem um valor em vários idiomas. Se o value e o valueId estiverem presentes, então, para cada string em um idioma específico, o significado deve ser o mesmo que o especificado em valueId (Plattform Industrie 4.0, 2023).

Figura 22 – Submodelo Multi Linguagem

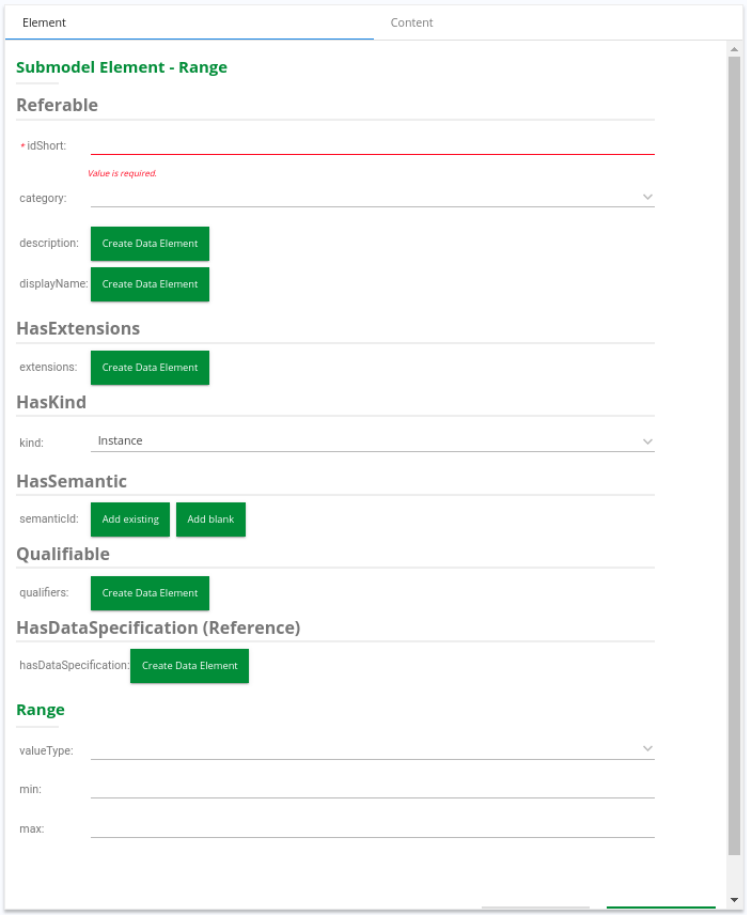



Fonte: (Elaborado pelo autor, 2023)

4.2.3.1.4 Intervalo (Range) A figura 23 representa o submodelo intervalo que é um elemento de dados definindo um intervalo com mínimo e máximo (Plattform Industrie 4.0, 2023).

- **min:** O valor mínimo do intervalo. Se o valor mínimo estiver ausente, o valor será assumido como infinito negativo.
- **max:** O valor máximo do intervalo. Se o valor máximo estiver faltando, então o valor será considerado infinito positivo.

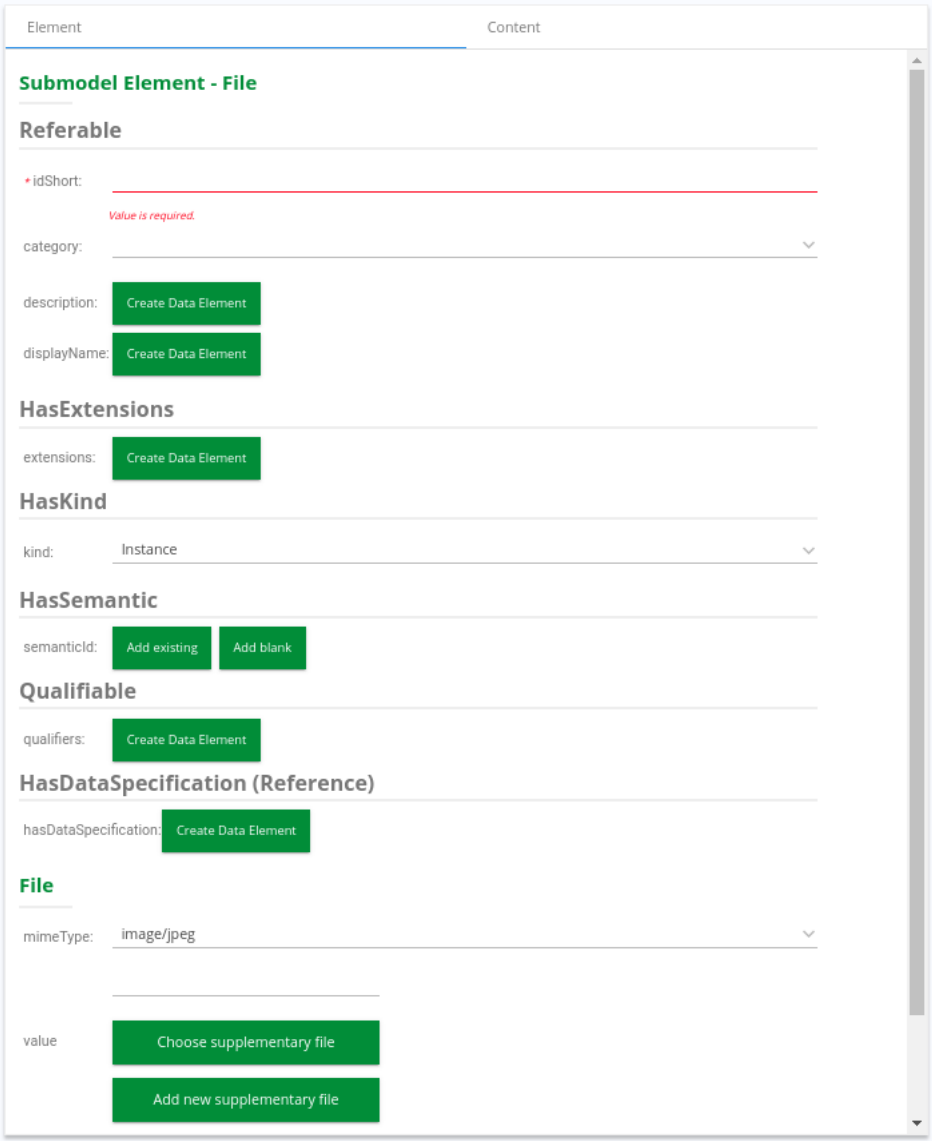
Figura 23 – Submodelo Intervalo




Fonte: (Elaborado pelo autor, 2023)

4.2.3.1.5 Arquivo (File) O Submodelo Arquivo, representado pela figura 24, é um elemento de dados que representa um endereço para um arquivo (localizador). O valor é um URI que pode representar um caminho absoluto ou relativo. Um caminho é usado no caso de o arquivo existir independentemente do *AAS*. O caminho relativo, relativo à raiz do pacote deve ser usado se o arquivo fizer parte do pacote serializado do *AAS* (Plattform Industrie 4.0, 2023).

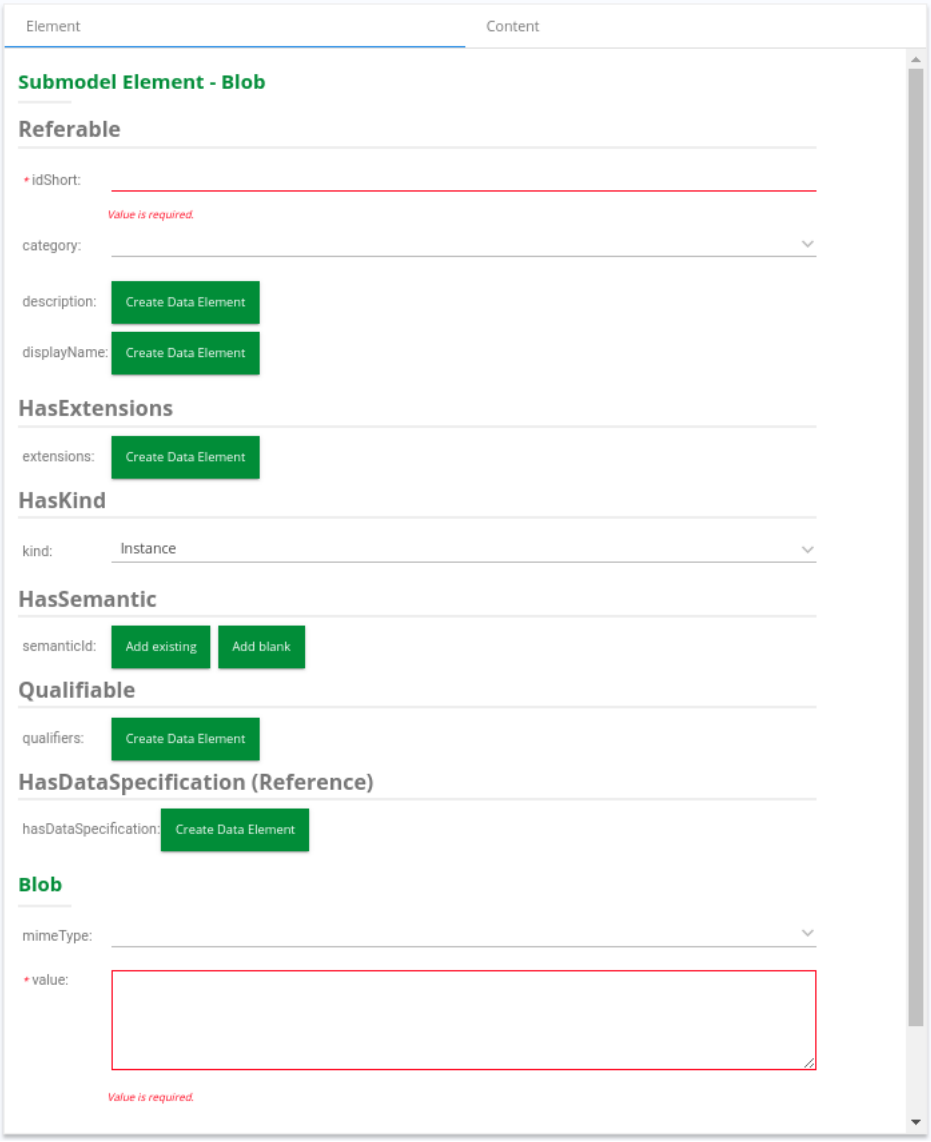
Figura 24 – Submodelo File




Fonte: (Elaborado pelo autor, 2023)

4.2.3.1.6 Blob A figura 25 representa, o submodelo blob, que é um elemento de dados que representa um arquivo contido com seu código-fonte no atributo value (Plattform Industrie 4.0, 2023).

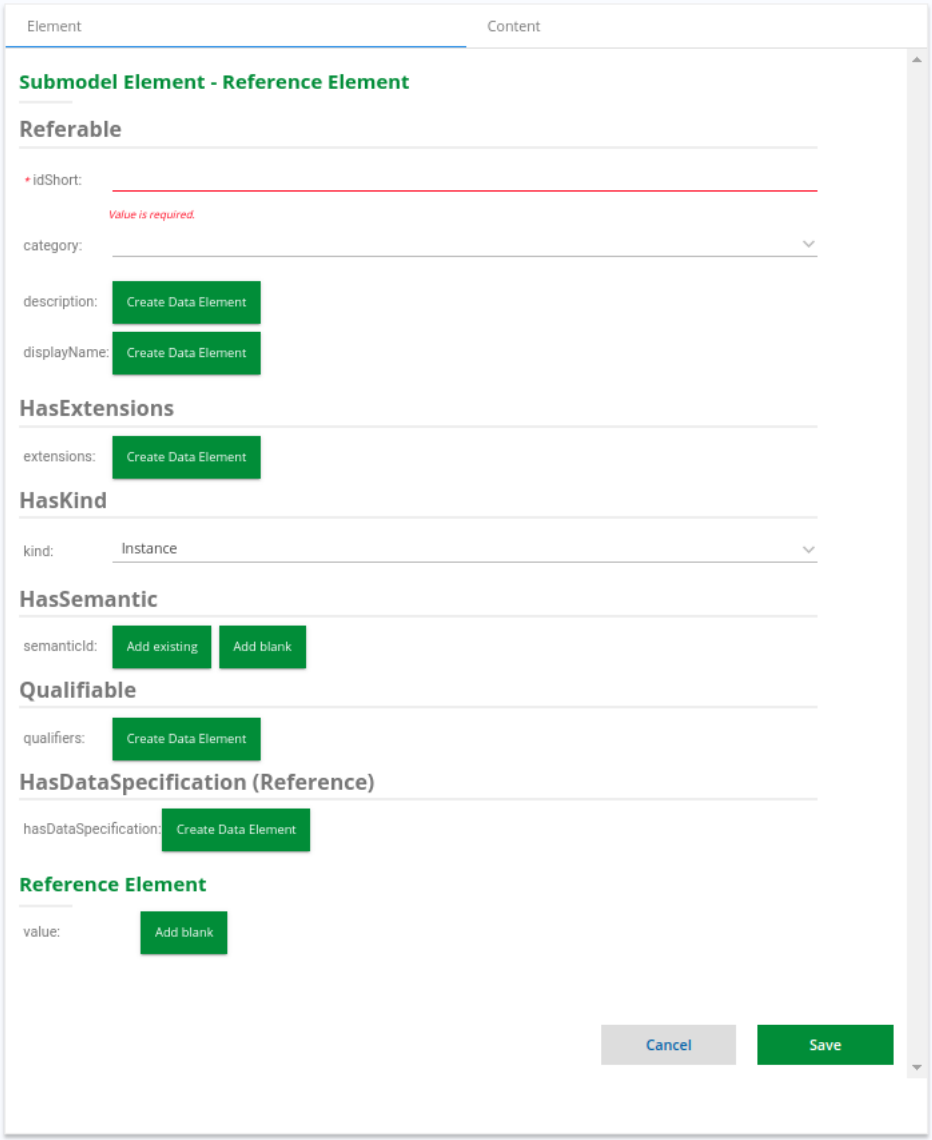
Figura 25 – Submodelo Blob




Fonte: (Elaborado pelo autor, 2023)

4.2.3.1.7 Referência (Reference Element) O submodelo de referência, demonstrado na figura 26, é um elemento de dados que define uma referência lógica a outro elemento dentro do mesmo, de outro *AAS* ou uma referência a um objeto ou entidade externa (Plattform Industrie 4.0, 2023).

Figura 26 – Submodelo Referência

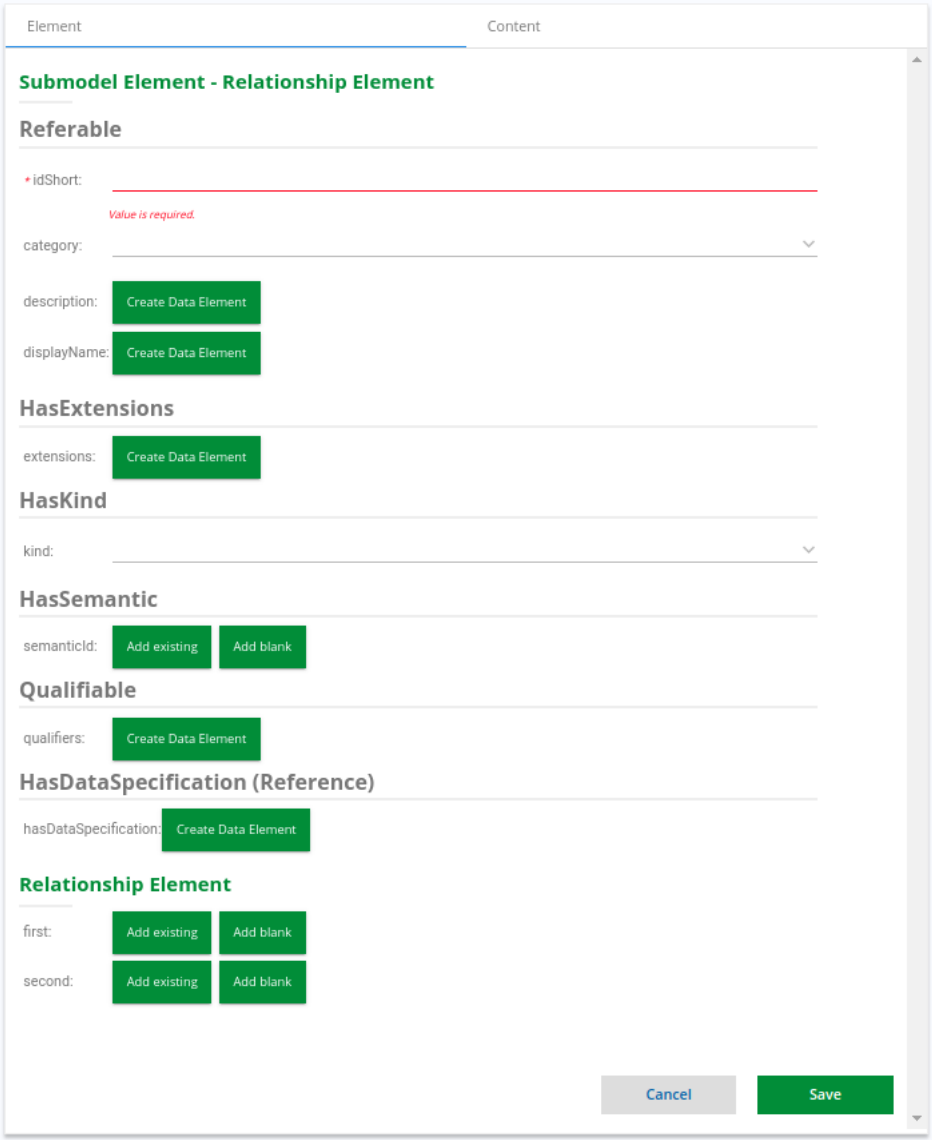



Fonte: (Elaborado pelo autor, 2023)

#### 4.2.3.1.8 Relacionamento (Relationship Element)

Este submodelo representado pela figura 27, é usado para definir um relacionamento entre dois elementos, sendo referenciável (referência de modelo) ou externo (referência global), (Plattform Industrie 4.0, 2023).

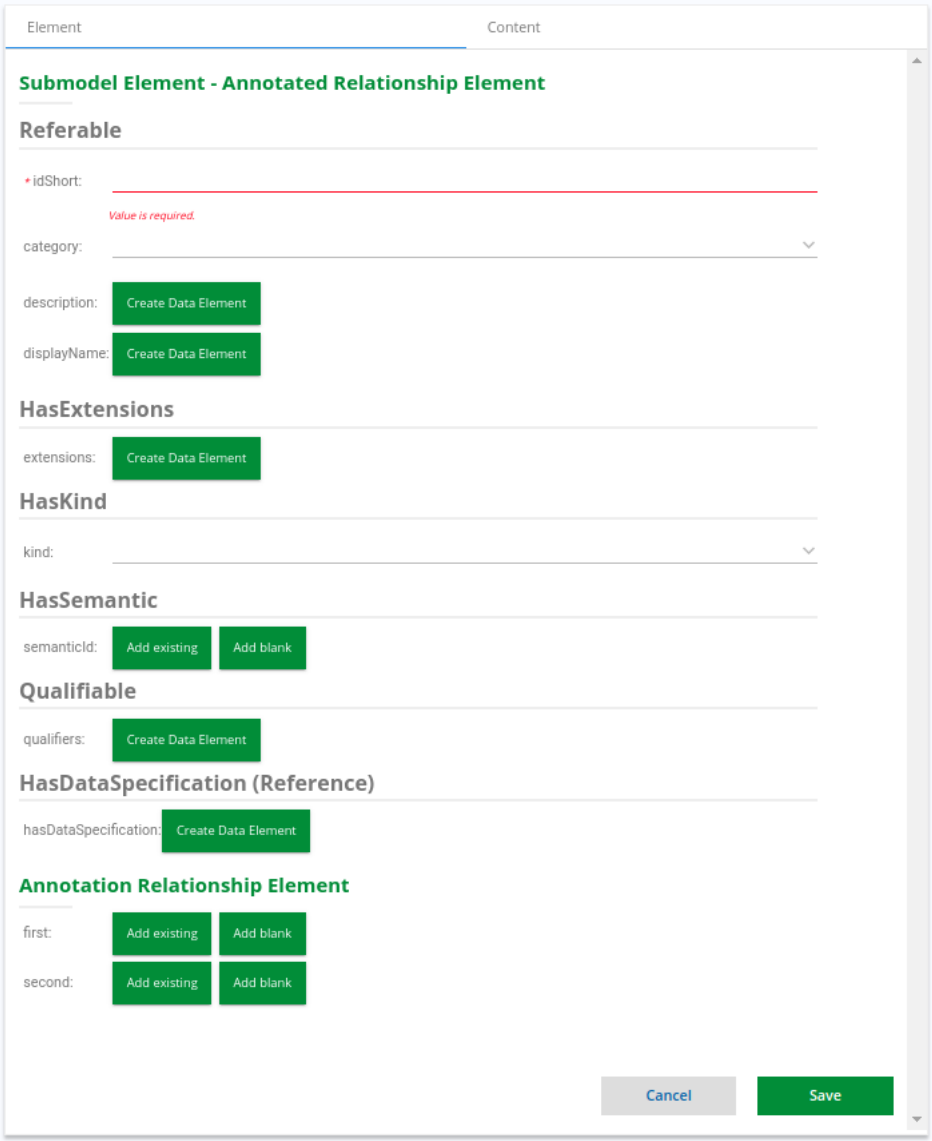
Figura 27 – Submodelo de Relacionamento




Fonte: (Elaborado pelo autor, 2023)

4.2.3.1.9 Relação Anotada (Annotated Relationship Element) A figura 28 representa a classe de relação anotada, um elemento de relação que pode ser anotado com elementos de dados adicionais. Além disso, representa uma anotação que se refere à relação entre os dois elementos (Plattform Industrie 4.0, 2023).

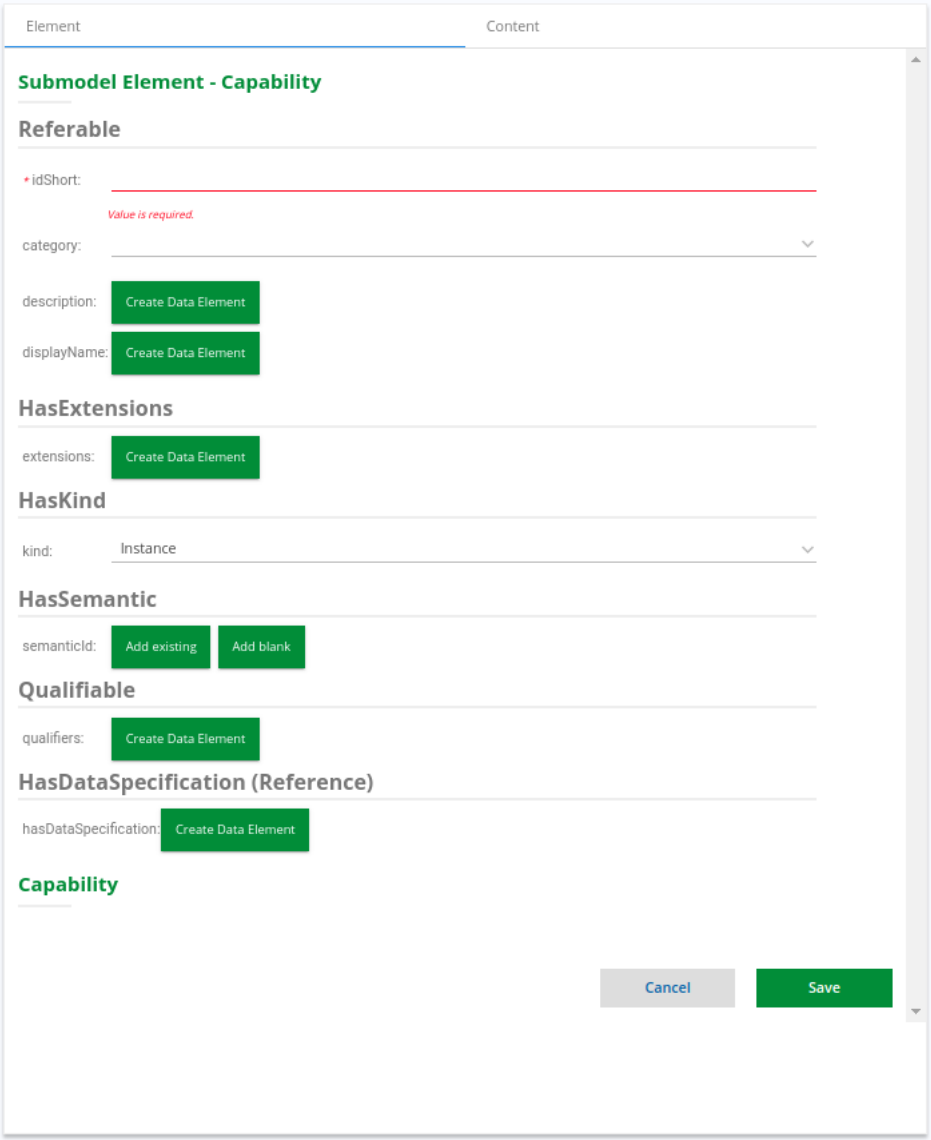
Figura 28 – Submodelo Relação Anotada




Fonte: (Elaborado pelo autor, 2023)

4.2.3.1.10 Capacidade (Capability) A figura 29 representa o submodelo capacidade, um recurso é a descrição independente da implementação do potencial de um ativo ativo para obter um determinado efeito no mundo físico ou virtual (Plattform Industrie 4.0, 2023).

Figura 29 – Submodelo Capacidade

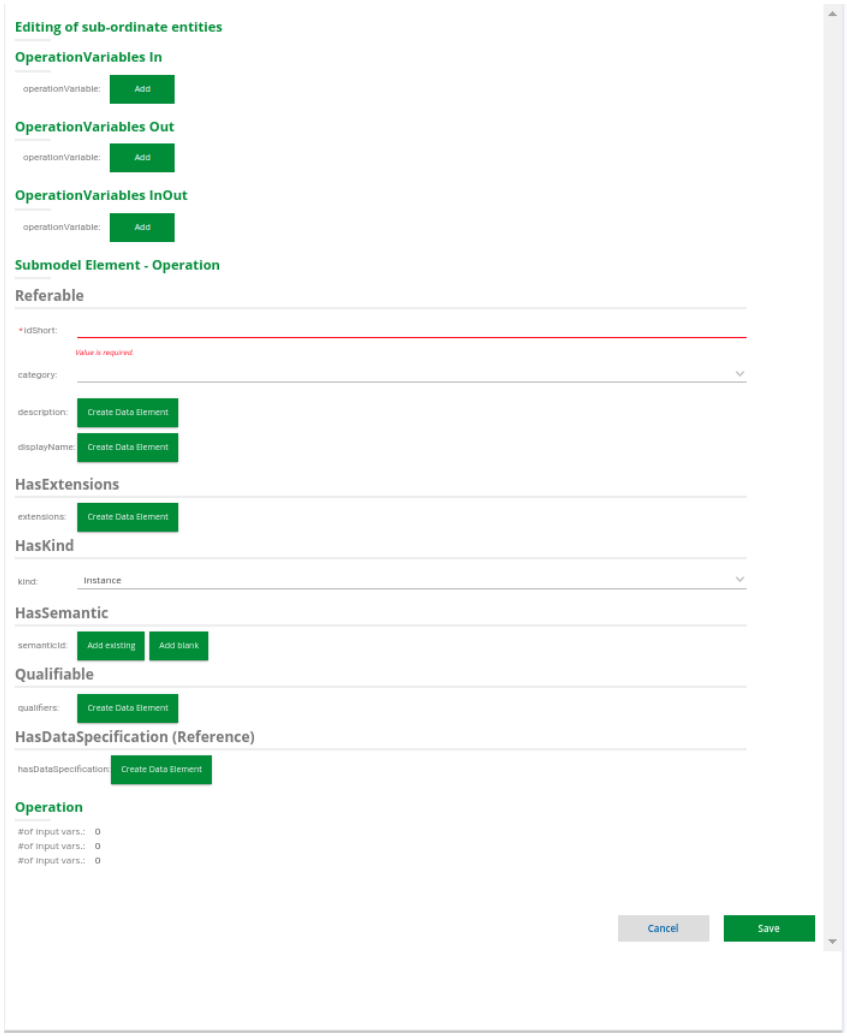



Fonte: (Elaborado pelo autor, 2023)

4.2.3.1.11 Operação (Operation) A figura 30 representa o submodelo operação, um elemento de submodelo com variáveis de entrada e saída (Plattform Industrie 4.0, 2023). Os atributos do submodelo operação:

- **inputVariable:** Parâmetro de entrada da operação.
- **outputVariable:** Parâmetro de saída da operação.
- **inoutputVariable:** Parâmetro que é a entrada e a saída da operação.

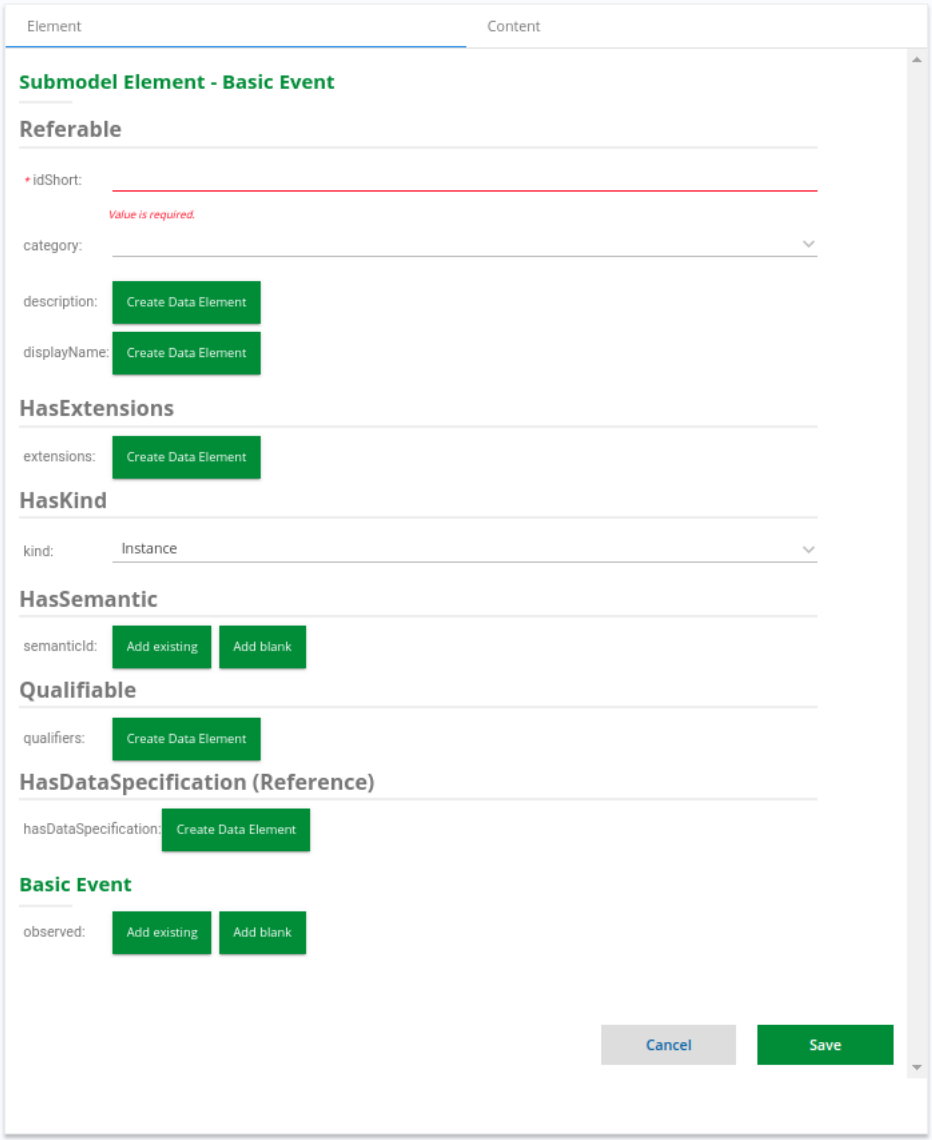
Figura 30 – Submodelo Operação




Fonte: (Elaborado pelo autor, 2023)

4.2.3.1.12 Evento Básico (Basic Event) A figura 31 representa o submodelo evento, este submodelo dentro de *AAS* define a estrutura e os relacionamentos dos vários elementos envolvidos na captura, no gerenciamento e na resposta a eventos relacionados a ativos (Plattform Industrie 4.0, 2023).

Figura 31 – Submodelo de Evento Básico

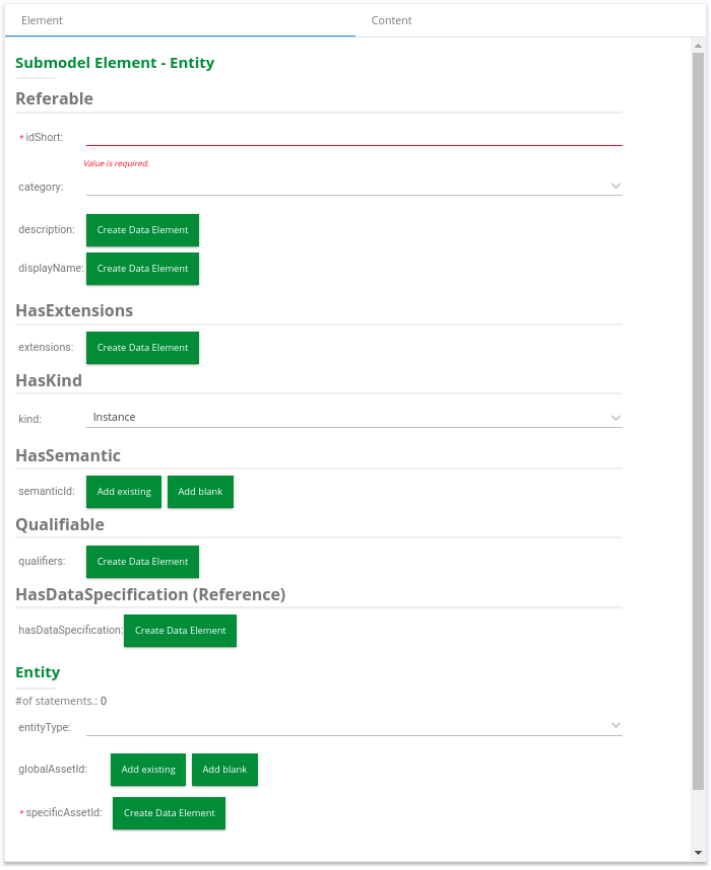



Fonte: (Elaborado pelo autor, 2023)

4.2.3.1.13 Entidade (Entity) A figura 32 representa o submodelo entidade. A entidade é um elemento de submodelo usado para modelar entidades. O atributo globalAssetId ou specificAssetId de uma entidade deve ser definido, se entityType for definido como "SelfManagedEntity". Eles não são existentes caso contrário (Plattform Industrie 4.0, 2023).

- **globalAssetId:** Identificador global do ativo que a entidade está representando.
- **specificAssetId:** Referência a um ID de ativo específico que representa um identificador suplementar do ativo representado pelo Shell de administração de ativos.
- **selfManagedEntity:** As entidades autogerenciadas têm sey próprio *AAS*, mas podem fazer parte do material de uma entidade autogerenciada composta.

Figura 32 – Submodelo Entidade

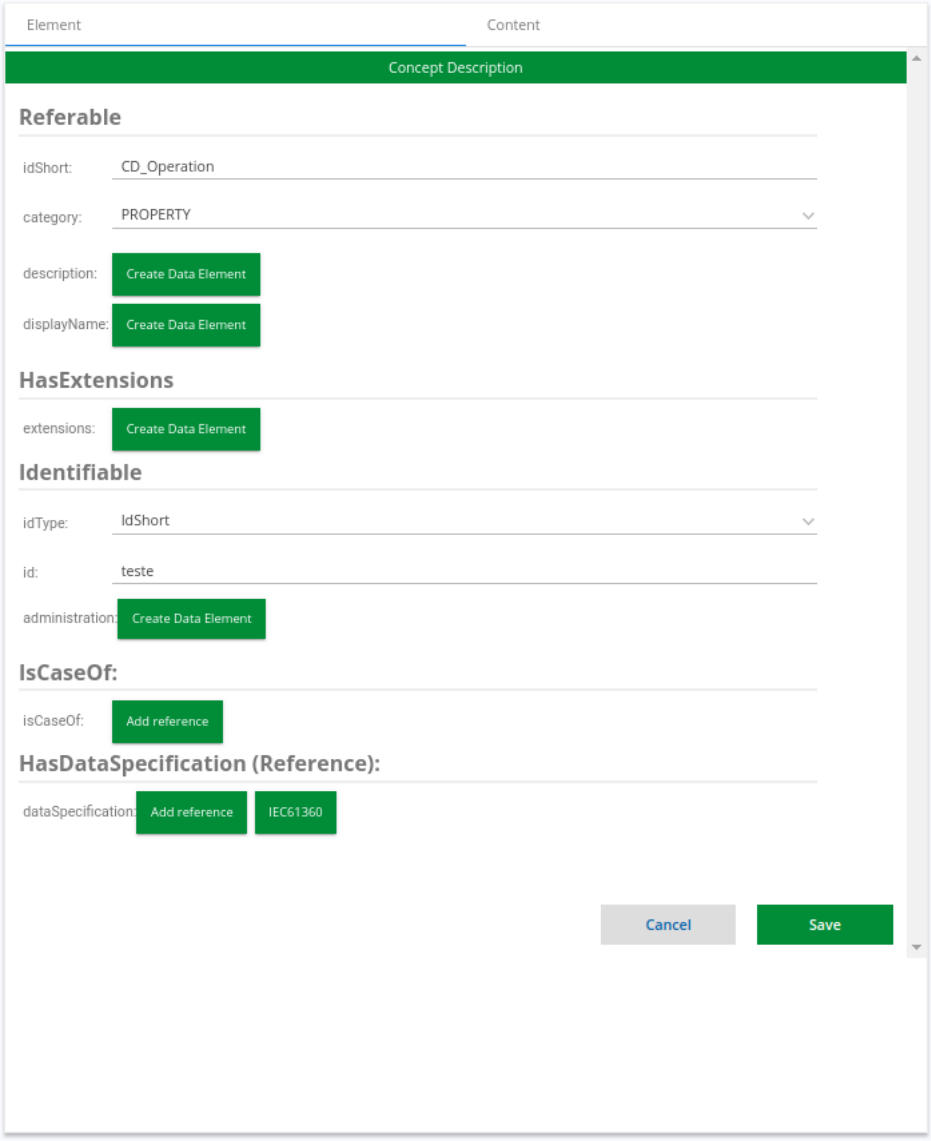



Fonte: (Elaborado pelo autor, 2023)

### 4.2.4 Descrição de Conceitos (Concept Description)

A semântica de uma propriedade ou de outros elementos que podem ter uma descrição semântica é definida por uma descrição de conceito representado pela figura 33. A descrição do conceito deve seguir um esquema padronizado, implementado como modelo de especificação de dados, (Plattform Industrie 4.0, 2023) .

Figura 33 – Descrição de Conceitos

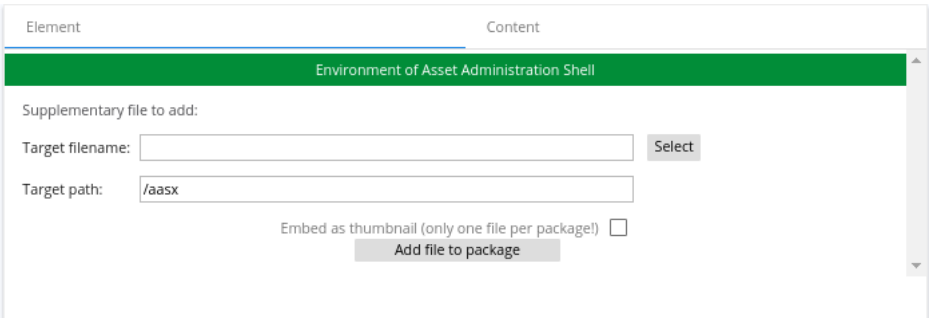



Fonte: (Elaborado pelo autor, 2023)

### 4.2.5 Arquivo suplementar (Supplementary File)

O metamodelo demonstrado pela figura 34, é utilizado para associar arquivos e documentos que são de utilidades dos operadores que acessam os dados referenciados pelo *Asset Administration Shell*.

Figura 34 – Arquivo suplementar




Fonte: (Elaborado pelo autor, 2023)

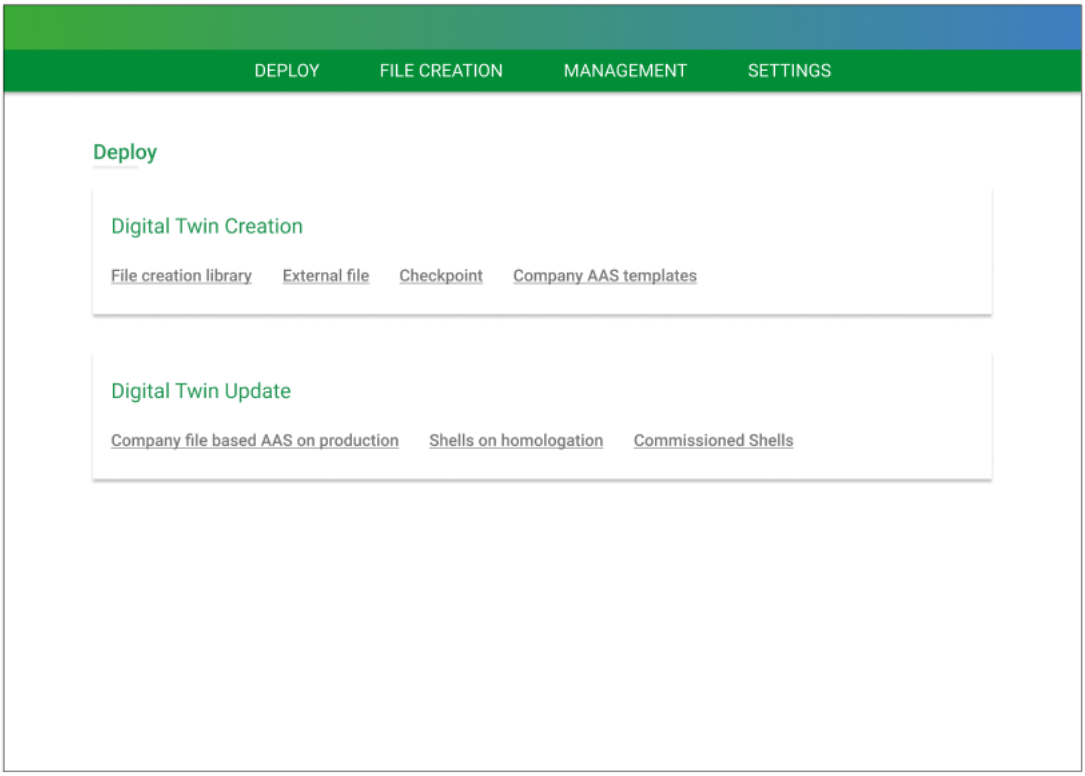
## 4.3 Deploy/Update de Shells

A figura 35 exibe a página inicial do módulo de deploy e update de Shells. A partir dela pode ser criado ou atualizado um gêmeo digital partindo de um arquivo base ou por meio de um template.

Nessa seção é possível alterar/atualizar os Shells baseados em arquivo e os Shells baseados em comunicação *OPC UA* que estão em homologação, realizar configurações, onde podemos escolher os servidores padrões de homologação e comissionamento de Shell. Por fim, também é possível configurar os submodelos básicos da empresa e templates de *Asset Administration Shell*.

Inicialmente é possível a implantar a partir de Shell modelados no módulo de criação que estão salvos na biblioteca de Shell ou através de arquivos externos com extensão (.aasx) que são reconhecidos normalmente pelo sistema. Além disso, ambas as formas seguem os mesmos passos de implantação e validação.

Figura 35 – Página inicial do módulo de deploy




Fonte: (Elaborado pelo autor, 2023)

### 4.3.1 Página de implantação de gêmeos digitais

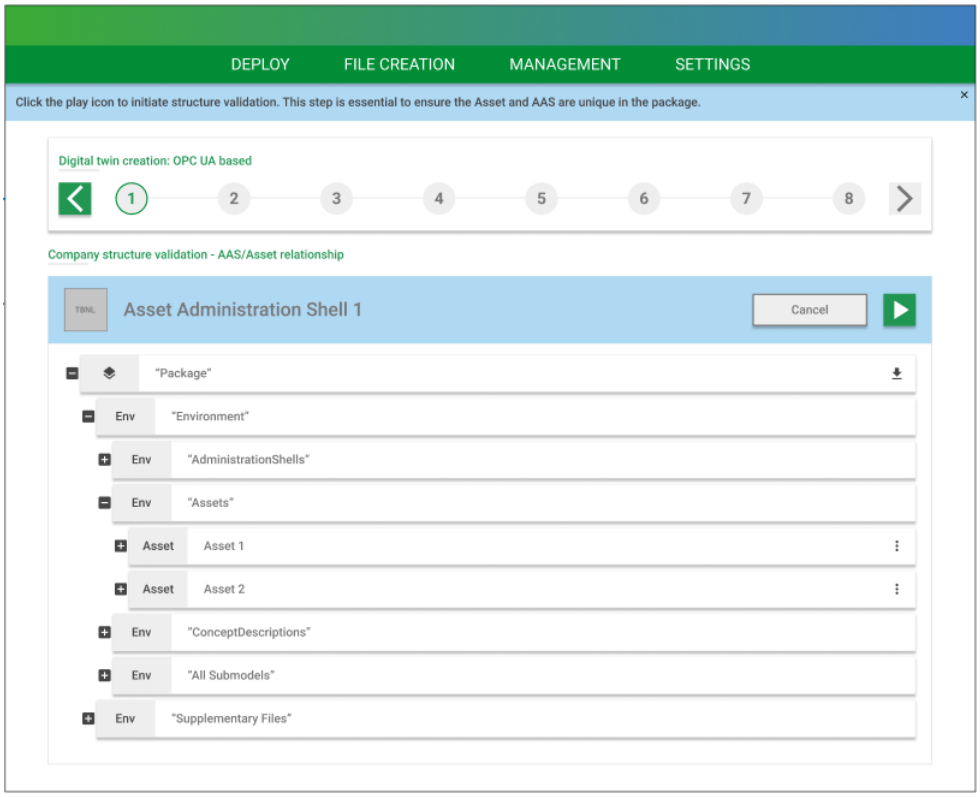
A implantação é divida em etapas. Essas etapas são definidas baseando-se no tipo de Shell escolhido na página inicial do módulo de deploy. Abaixo é demonstrado as etapas e suas respectivas valições.

#### 4.3.1.1 Passo 1: Validação da estrutura e relacionamento

No passo 1 demonstrado na figura 36, o operador tem a visualização da estrutura do pacote escolhido e pode iniciar o processo de deploy. Esse passo verifica se há apenas um *AAS* e um Asset no pacote, ou seja, um relacionamento (1x1), se houver mais de um Asset por *AAS*, o operador terá que realizar os ajustes necessários. O usuário só poderá prosseguir nos passos se completar o(s) anterior(es) devidamente.

É importante ressaltar que na implatanção e atualização de Shell esse passo é crucial para evitar a inconsistência de pacotes e duplicação de Assets.

Figura 36 – Tela de validação da estrutura e relacionamento



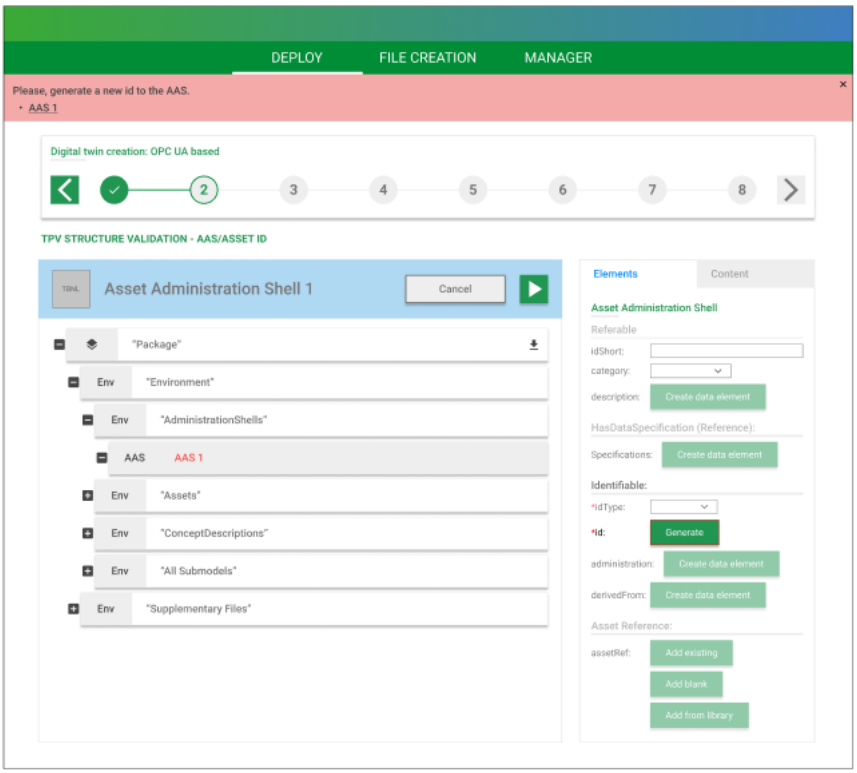

Fonte: (Elaborado pelo autor, 2023)

#### 4.3.1.2 Passo 2: Validação de IDs

O segundo passo, demonstrado na figura 37, consiste na checagem dos identicadores únicos do *AAS* e do Asset para assegurar que não há outros componentes no sistema com os mesmos IDs.

Caso o operador se depare um ID já utilizado, a interface o possibilita gerar um novo ID sem a necessidade ter que retornar ao módulo de criação para realizar as alterações necessárias.

Figura 37 – Tela de alidação de IDs

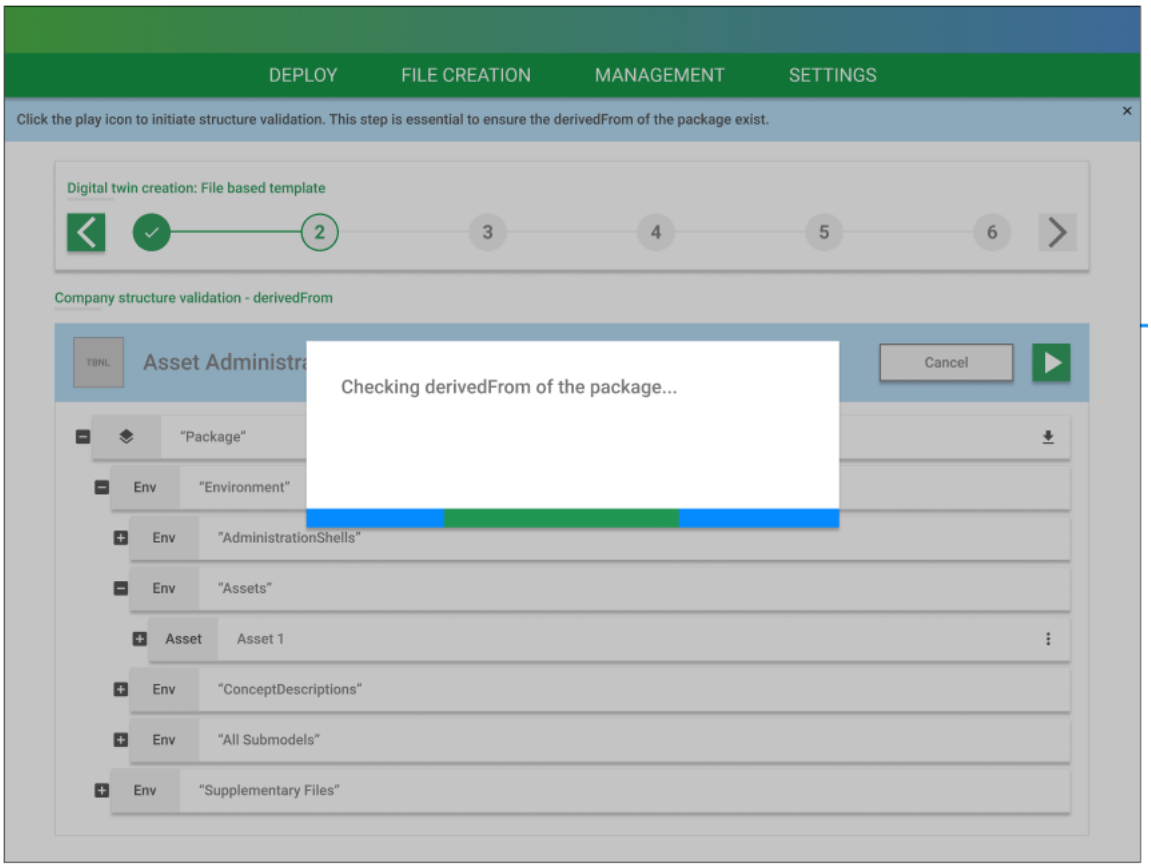



Fonte: (Elaborado pelo autor, 2023)

#### 4.3.1.3 Passo 2: Validação de template (Shells templates)

Os *Asset Administration Shell* baseados em templates são casos específicos de implantação e atualização, possuem um passo extra de validação. Nesse caso o passo 2 demonstrado na figura 38, consiste na checagem em qual template a instância de *AAS* está sendo criada. Caso não haja a propriedade derivedFrom ou o derivedFrom aponte para um template não conhecido pelo sistema, essa etapa retornará um erro.

Figura 38 – Tela Validação de template




Fonte: (Elaborado pelo autor, 2023)

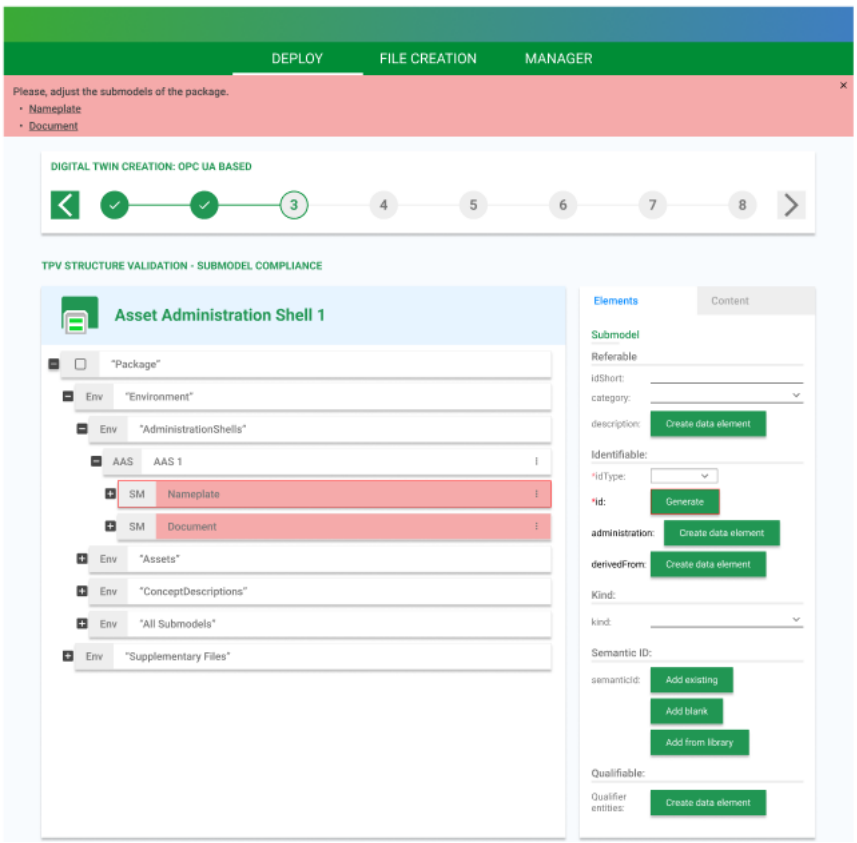
#### 4.3.1.4 Passo 3: Validação complaince

O passo 3, demonstrado na figura 39, consiste em garantir a conformidade com as especificações de *AAS* definidas pela empresa. Além disso implica que um submodelo dentro de um *AAS* deve atender aos padrões e requisitos específicos estabelecidos para a criação e gestão de submodelos.

A conformidade de submodelos é crucial para assegurar que as informações sobre os ativos sejam estruturadas e formatadas de uma maneira que possa ser compreendida e utilizada por outros sistemas e equipamentos que interagem com o *AAS*.

Caso o operador se depare com essa inconsistência, deverá retorna a módulo de criação e revaliar as propriedades dos submodelos.

Figura 39 – Tela de validação complaince

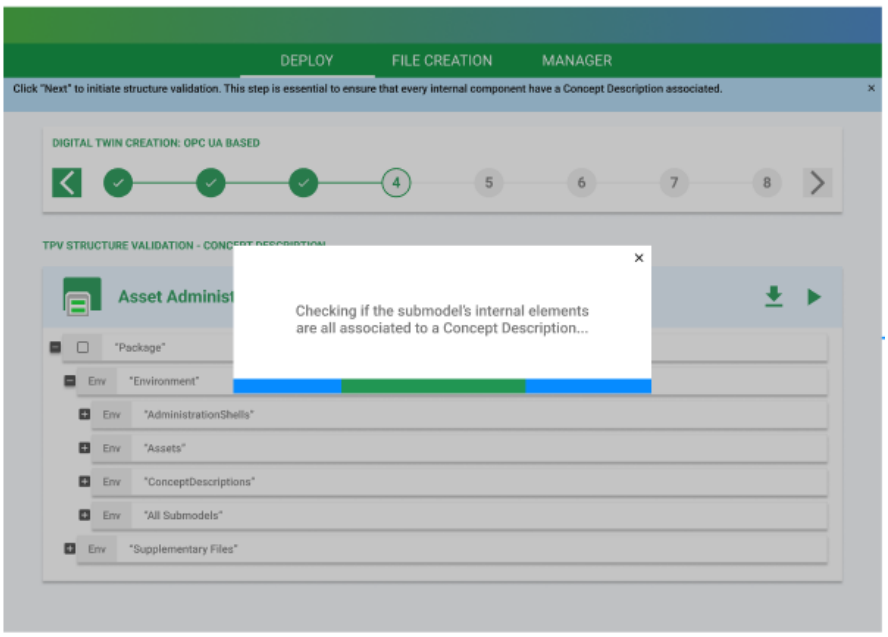



Fonte: (Elaborado pelo autor, 2023)

#### 4.3.1.5 Passo 4: Validação de descrições de conceito

O passo 4, demonstrado na figura 40, é um processo essencial para garantir que a descrição do conceito esteja correta, precisa e alinhada com os padrões e requisitos estabelecidos. As descrições de conceitos podem incluir informações sobre as funcionalidades, interfaces, propriedades e comportamentos dos ativos. Caso o operador se depare com essa inconsistência, deverá retorna a módulo de criação.

Figura 40 – Tela de validação de descrições de conceito



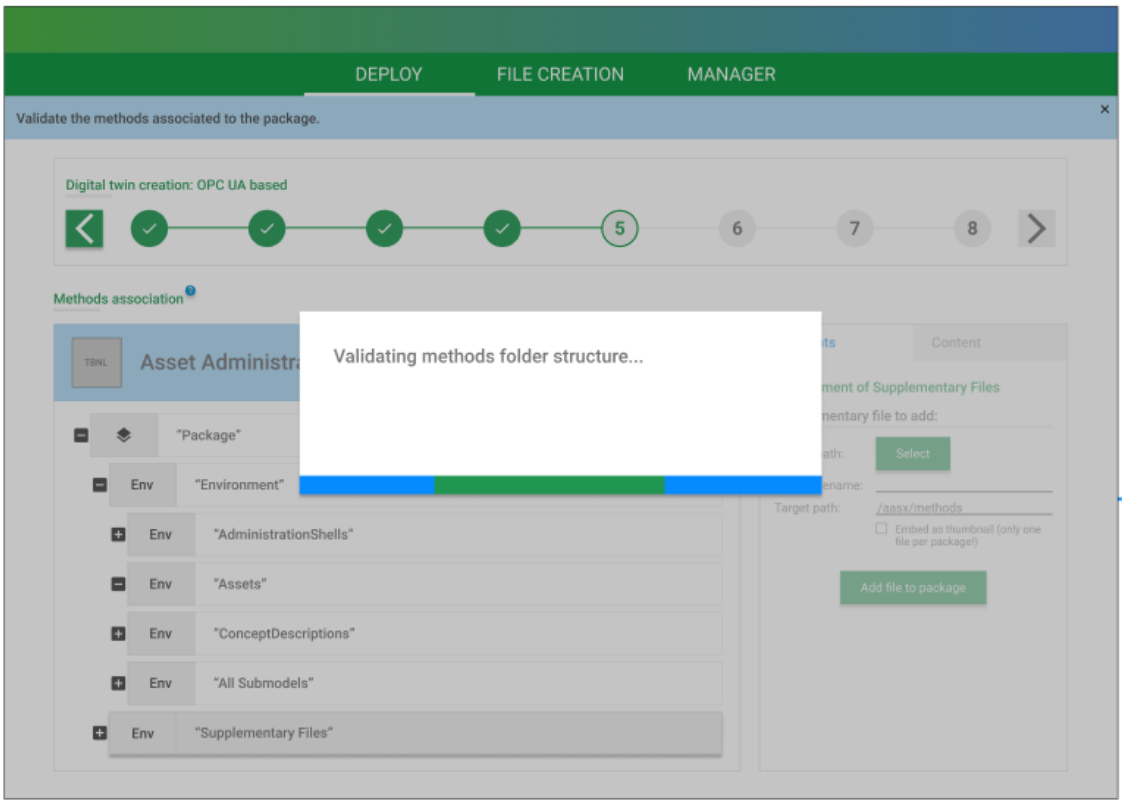

Fonte: (Elaborado pelo autor, 2023)

#### 4.3.1.6 Passo 5: Validação da associação de métodos

O passo 5, demonstrado na figura 41, consiste na validação do arquivo de método implementados para o *AAS*, isso inclui a estrutura de pasta e possibilidade de métodos não implementados.

A associação de métodos a um *AAS* é uma maneira de fornecer funcionalidades e capacidades operacionais aos ativos digitais representados no ambiente industrial. Além disso a possibilita ao operador a geração de relatórios com informações sobre o estado ou desempenho dos ativos, bem como a análise de dados para tomar decisões informadas.

Figura 41 – Tela de validação de descrições de conceito




Fonte: (Elaborado pelo autor, 2023)

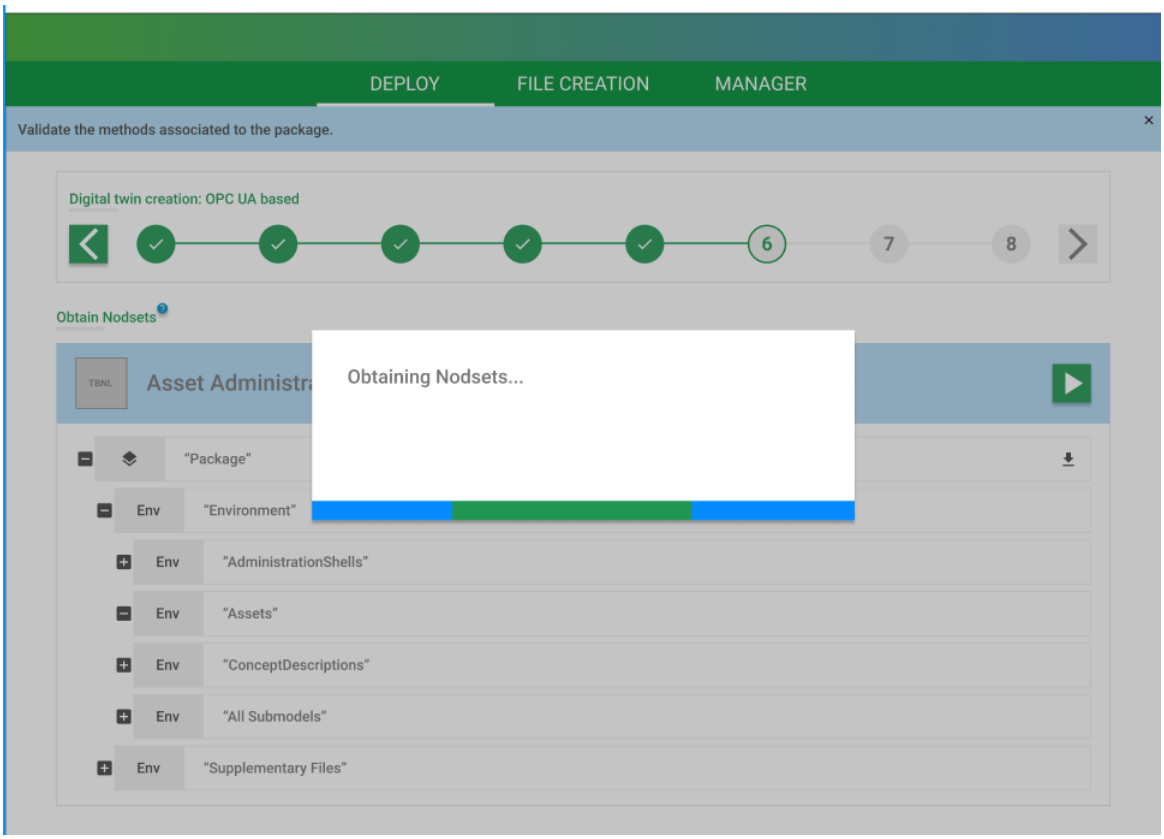
#### 4.3.1.7 Passo 6: Validação de NodeSets

O passo 6, demonstrado na figura 42, consiste na validação de NodeSets cuja função é permitir a organização hierárquica de elementos dentro de um *Asset Administration Shell*. Isso facilita a representação de ativos e suas características de forma estruturada.

Eles são usados para representar ativos físicos, virtuais ou conceituais, bem como seus componentes individuais, submodelos, interfaces, e outras entidades relacionadas. Isso inclui como os ativos se relacionam entre si, como os submodelos se conectam aos ativos, e assim por diante.

Com esse passo validado, os nodeSets possibilitam acesso a dados e funcionalidades associadas aos ativos representados pelo *AAS* como informações, o estado operacional, configurações, métodos disponíveis, e outros.

Figura 42 – Tela de validação de NodeSets

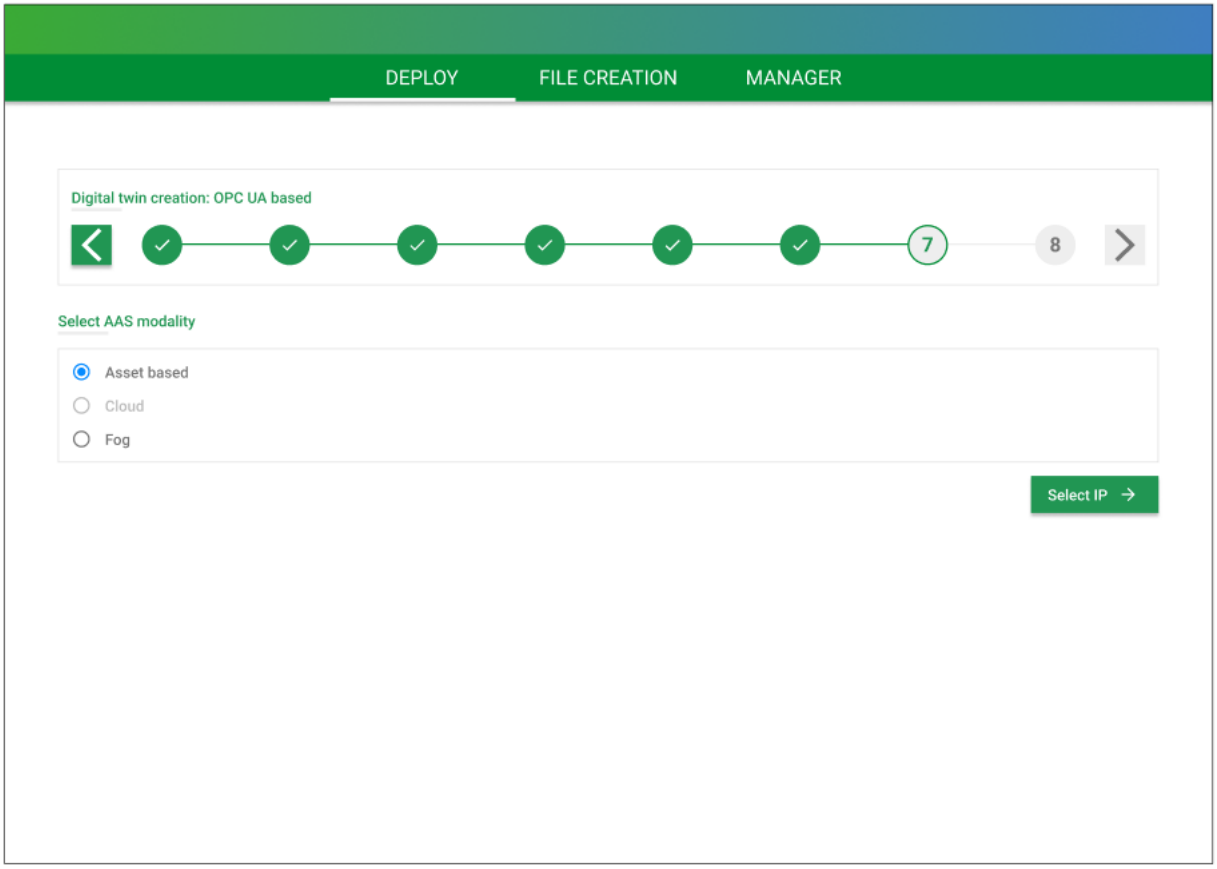



Fonte: (Elaborado pelo autor, 2023)

#### 4.3.1.8 Passo 7: Escolha de Modalidade

No passo 7, demonstrado na figura 43, o operador pode eleger o tipo de implantação para o respectivo *Asset Administration Shell*. De forma geral o operador informará ao sistema se o *AAS* terá um servidor específico e dedicado, ou se o servidor padrão será usado para implantação.

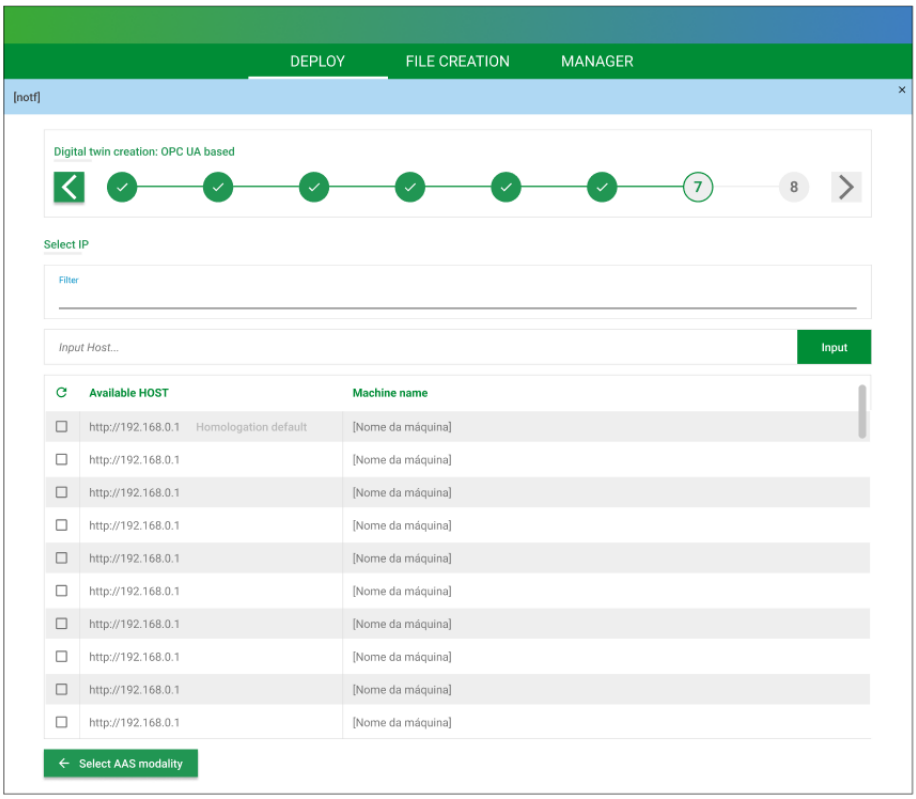
Figura 43 – Tela de Escolha de Modalidade




Fonte: (Elaborado pelo autor, 2023)

Caso o operador opte por usar implantação específica, será necessário eleger o respectivo servidor através da lista de host disponíveis ou do input como é demonstrado da figura 44.

Figura 44 – Tela de Escolha de servidor

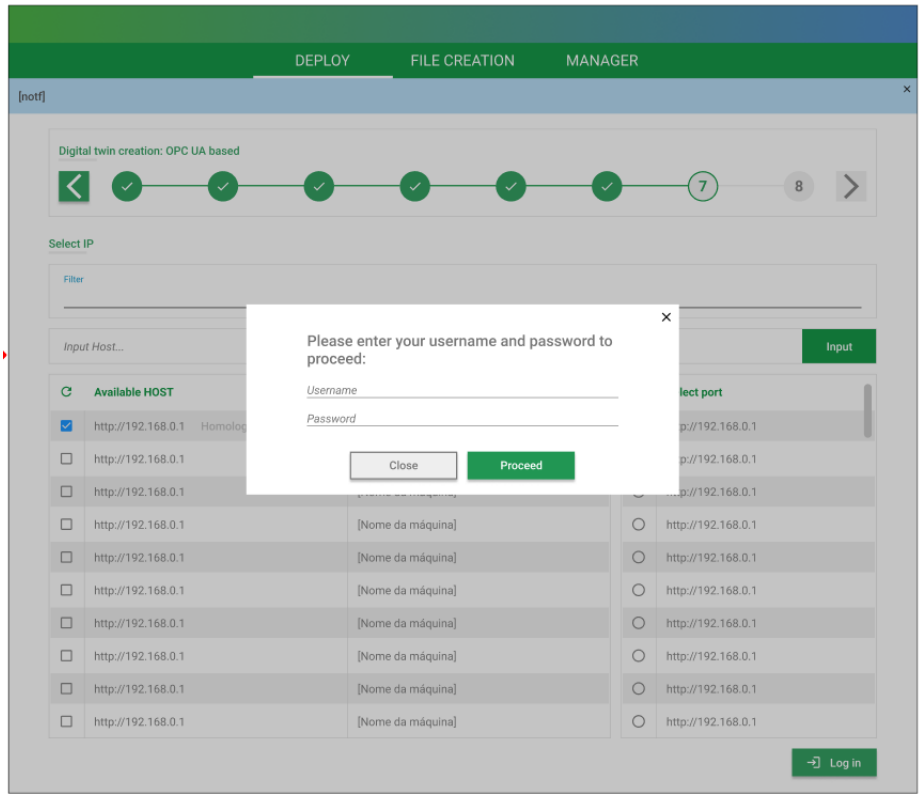



Fonte: (Elaborado pelo autor, 2023)

Após a escolha do host, o operador precisa inserir as crendeciais de acesso do servidor demonstrado na figura 45, para que as configuração e implantação seja realizada.

Caso haja erro ou sucesso na implantação o sistema emitirá alerta como feedback ao operador.

Figura 45 – Modal de inserção de credenciais

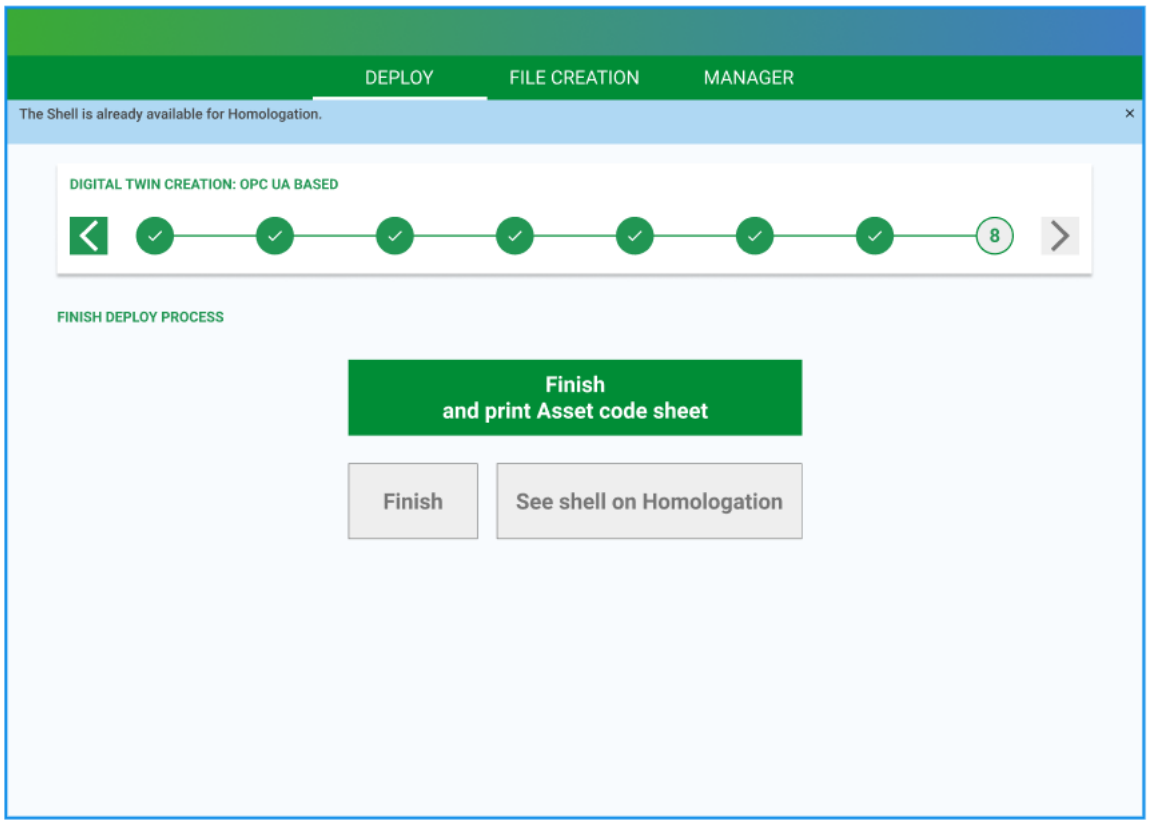



Fonte: (Elaborado pelo autor, 2023)

#### 4.3.1.9 Passo 8: Finalização e Acesso

No passo 8, demonstrado na figura 46, o operador pode eleger o tipo de implantação para o respectivo *Asset Administration Shell*. De forma geral o operador informará ao sistema se o *AAS* terá um servidor específico e dedicado, ou se o servidor padrão será usado para implantação.

Figura 46 – Tela de finalização e impressão




Fonte: (Elaborado pelo autor, 2023)

## 4.4 Gerenciamento de Shells

Esta seção abrange o módulo de genrenciamento de Shells, e sua principais características desenvolvidas. O gerenciamento eficaz de *Asset Administration Shell*s facilita a interoperabilidade entre diferentes ativos e sistemas dentro de ambientes da *Indústria 4.0*, significando que equipamentos de diferentes fabricantes e com protocolos de comunicação diferentes podem colaborar de forma eficiente.

A capacidade de gerenciar *Asset Administration Shell*s permite a implementação de estratégias de manutenção preditiva e preventiva, possibilitando que a manutenção possa ser realizada antes que ocorram falhas, reduzindo o tempo de inatividade e os custos.

Além disso, permite a tomada de decisões informadas com base em informações atualizadas sobre o estado dos ativos e dos processos de produção.

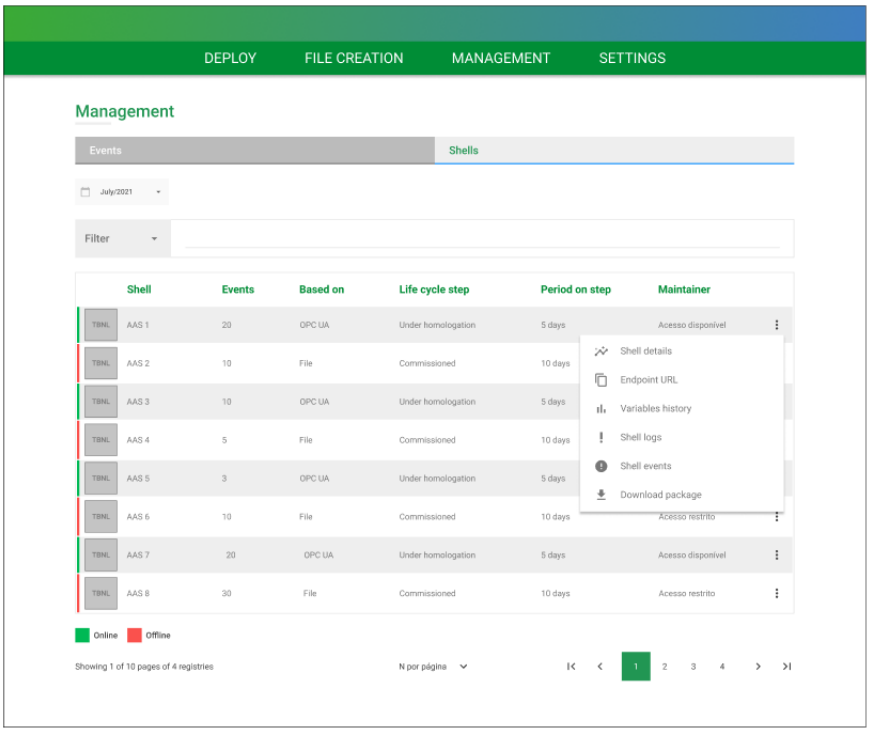
Algumas das principais caracteríticas abordadas são Monitoramento de logs, eventos e histórico de variáveis.

### 4.4.1 Lista de Shells

A página de monitoramento representada, na figura 47 abaixo é utilizada para o monitoramento dos Shell. Nela é possível visualizar o status atual do Shell que foi implantado, se o gêmeo digital estivar online recebe a cor verde, caso contrário vermelho.

Além disso, cada Shell listado possui ações como visualizar histórico de variáveis, eventos, logs, download do pacote implantado e detalhes do pacote.

Figura 47 – Tela de Monitoramento de Shells




Fonte: (Elaborado pelo autor, 2023)

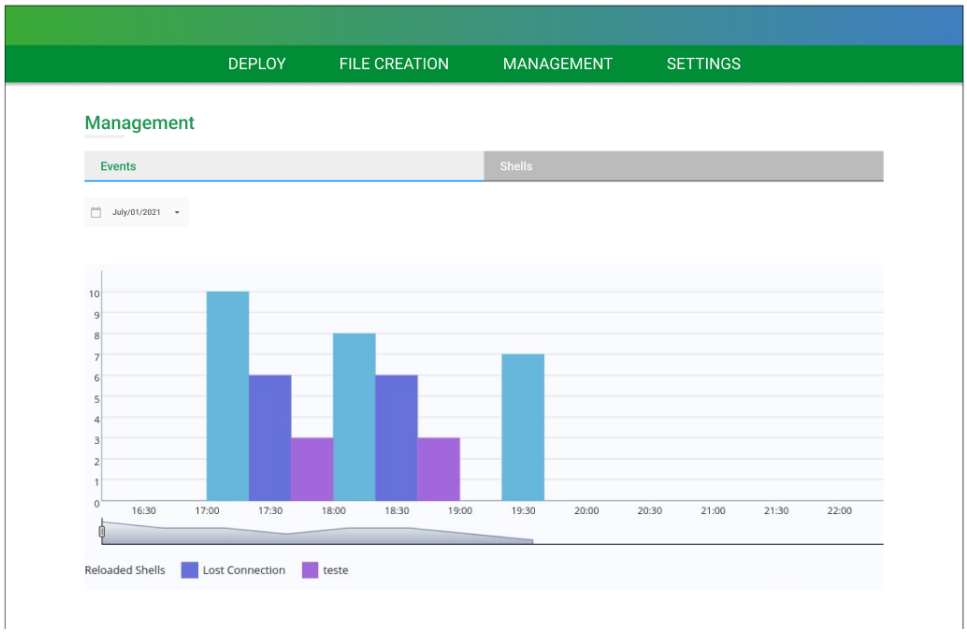
### 4.4.2 Eventos

A página de eventos representada nas figuras 48 e 50 abaixo é utilizada para o monitoramento e visualização de eventos.

Os eventos são notificações ou ocorrências que podem ser geradas por ativos ou sistemas para informar sobre mudanças de estado, condições ou eventos relevantes. Eles desempenham um papel fundamental na monitorização e gestão de ativos digitais.

No gráfico de barras representado na figura 48, é possível visualizar métricas dos eventos que acontenceram em determinado intervalo de tempo agrupadas por hora, filtrar intervalos específicos de tempo para realizar comparações e tomar decições com base nos eventos.

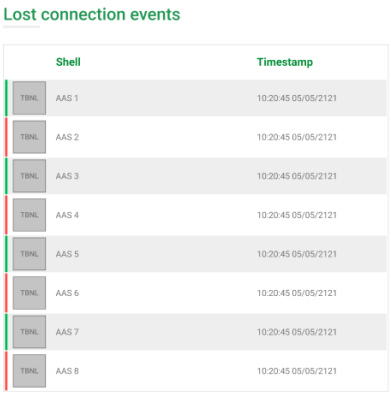
Figura 48 – Gráfico de monitoramento de eventos




Fonte: (Elaborado pelo autor, 2023)

Além disso, ao realizar a ação de clique em umas das barras é abertos um modal com uma lista todos os eventos que ela representa demonstrado na figura 49.

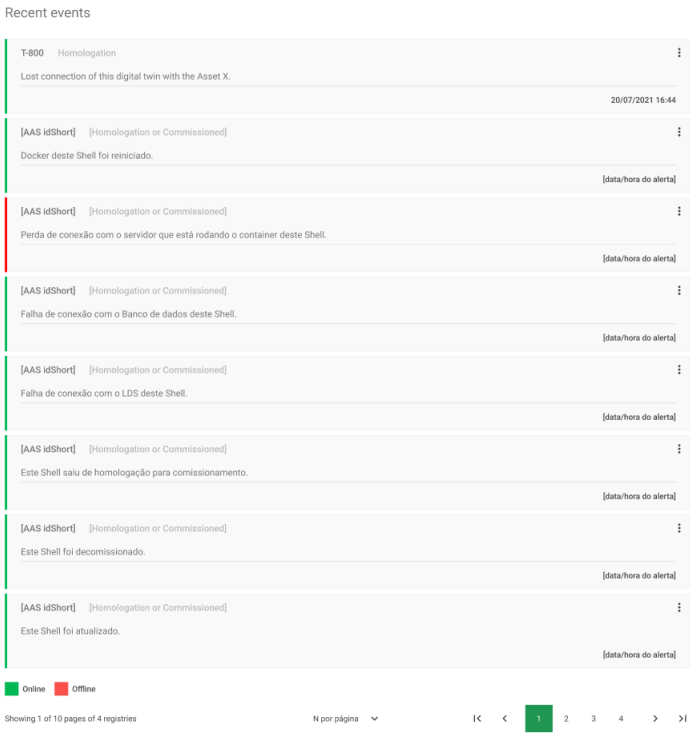
Figura 49 – Modal de eventos ao clicar na barra




Fonte: (Elaborado pelo autor, 2023)

Na parte inferior da tela de eventos há uma tabela com cards representada pela figura 50. Nela contém dados eventos recentes como data do ocorrido, status do Shell, tipo de evento e mensagem.

Figura 50 – Tablela de eventos recentes

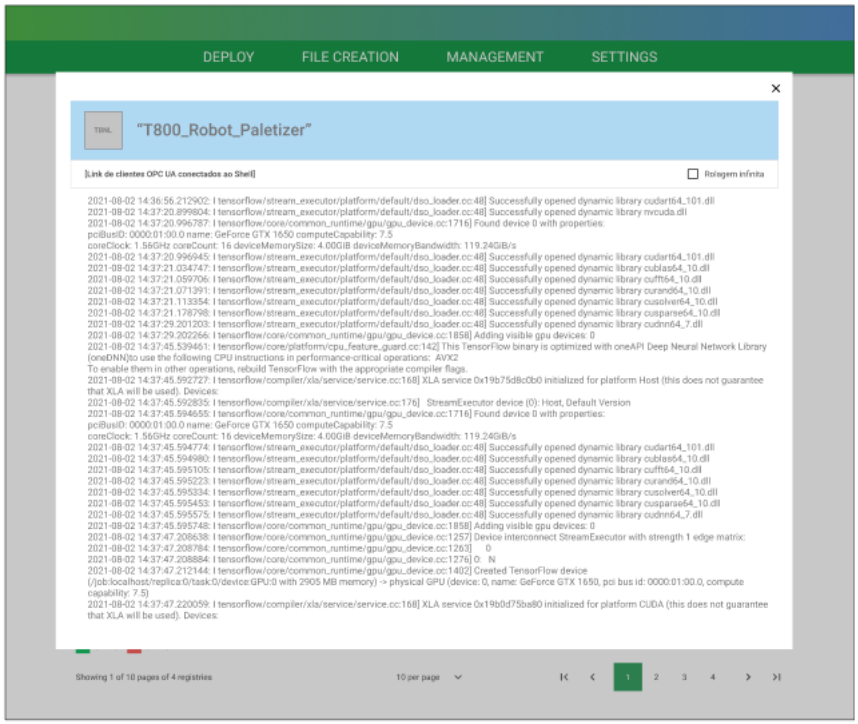



Fonte: (Elaborado pelo autor, 2023)

### 4.4.3 Logs

Na página de logs acessada pela listagem de Shells 4.4.1, o operador pode visualizar logs de cada Shell. Além disso, os logs podem ser monitorados em tempo real e os antigos podem ser visualizados através histórico.

Figura 51 – Modal de Eventos



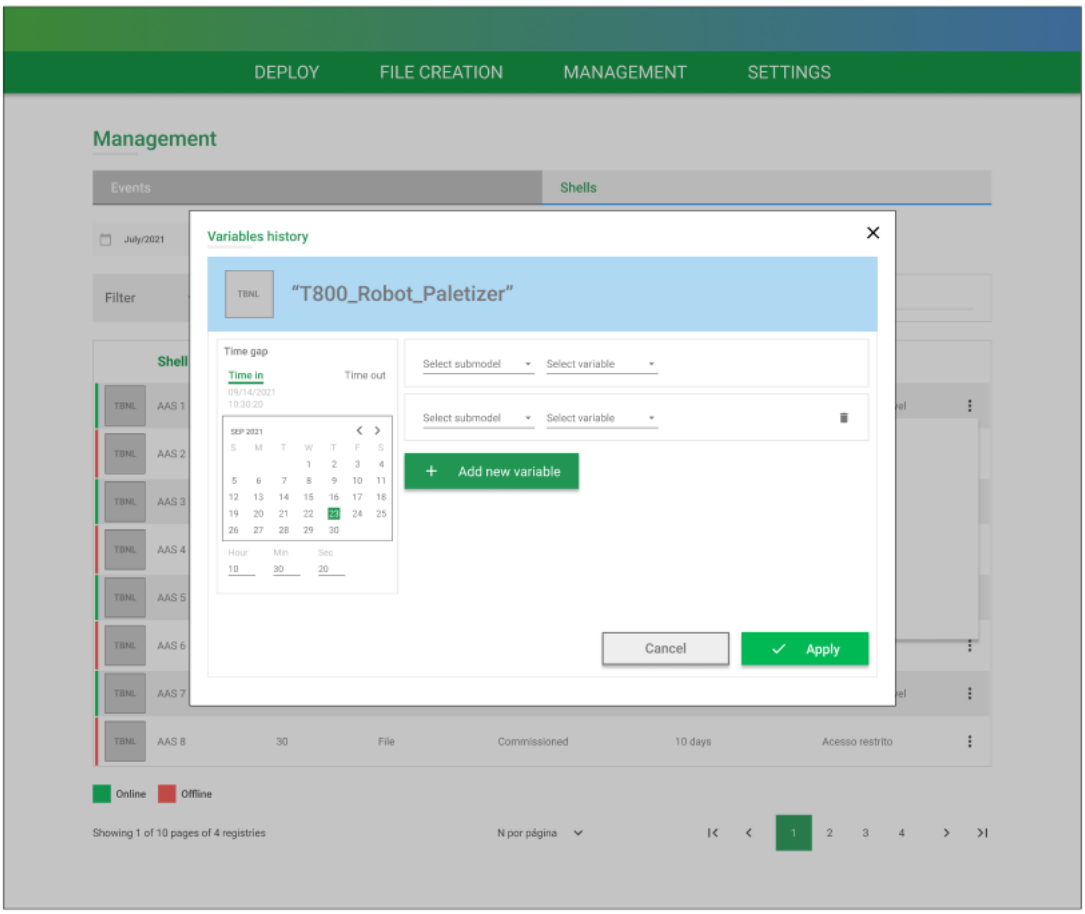

Fonte: (Elaborado pelo autor, 2023)

### 4.4.4 Histórico de variáveis

Na página de histórico de variáveis, acessada pela listagem de Shells 4.4.1, o operador pode visualizar o histórico de propriedades armazenados nos bancos de dados dos Shell (variáveis históricizadas). Como demostrado na figura 52 é possível selecionar um intervalo de tempo específico de dados armazenados, isso possibilita identificar padrões que indicam possíveis falhas iminentes em um ativo, permitindo agendar a manutenção antes que ocorram problemas graves.

Além disso, o operador pode aproveitar os benefícios da análise retrospectiva para aprimorar a eficiência operacional, a confiabilidade dos ativos e a tomada de decisões informadas. Essa funcionalidade é fundamental no desenvolvimento deste projeto.

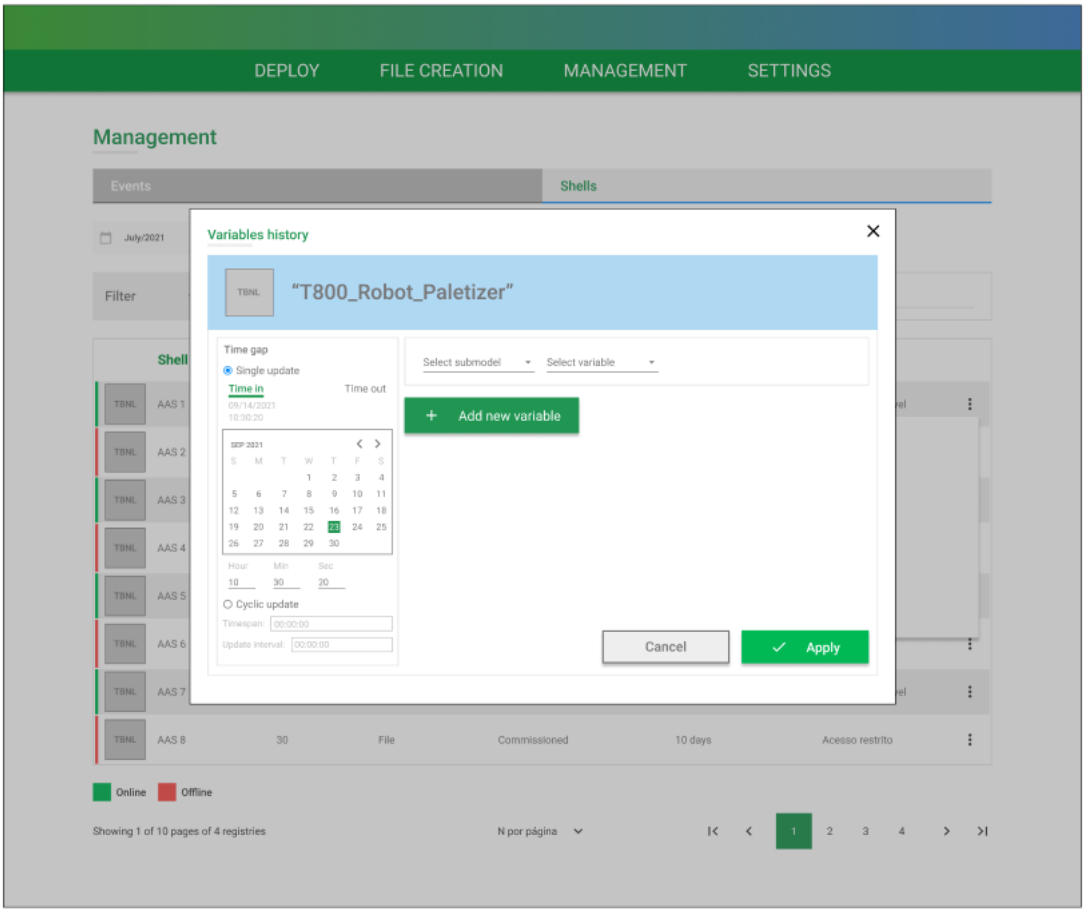
Figura 52 – Modal de seleção de intervalo de dados




Fonte: (Elaborado pelo autor, 2023)

O modal também permite a seleção de dados cíclios ou em tempo real mostrado na figura 53. Para isso é necessário selecionar a opção "ciclic update"e as variáveis a serem monitoradas.

Figura 53 – Modal de seleção dados cíclicos

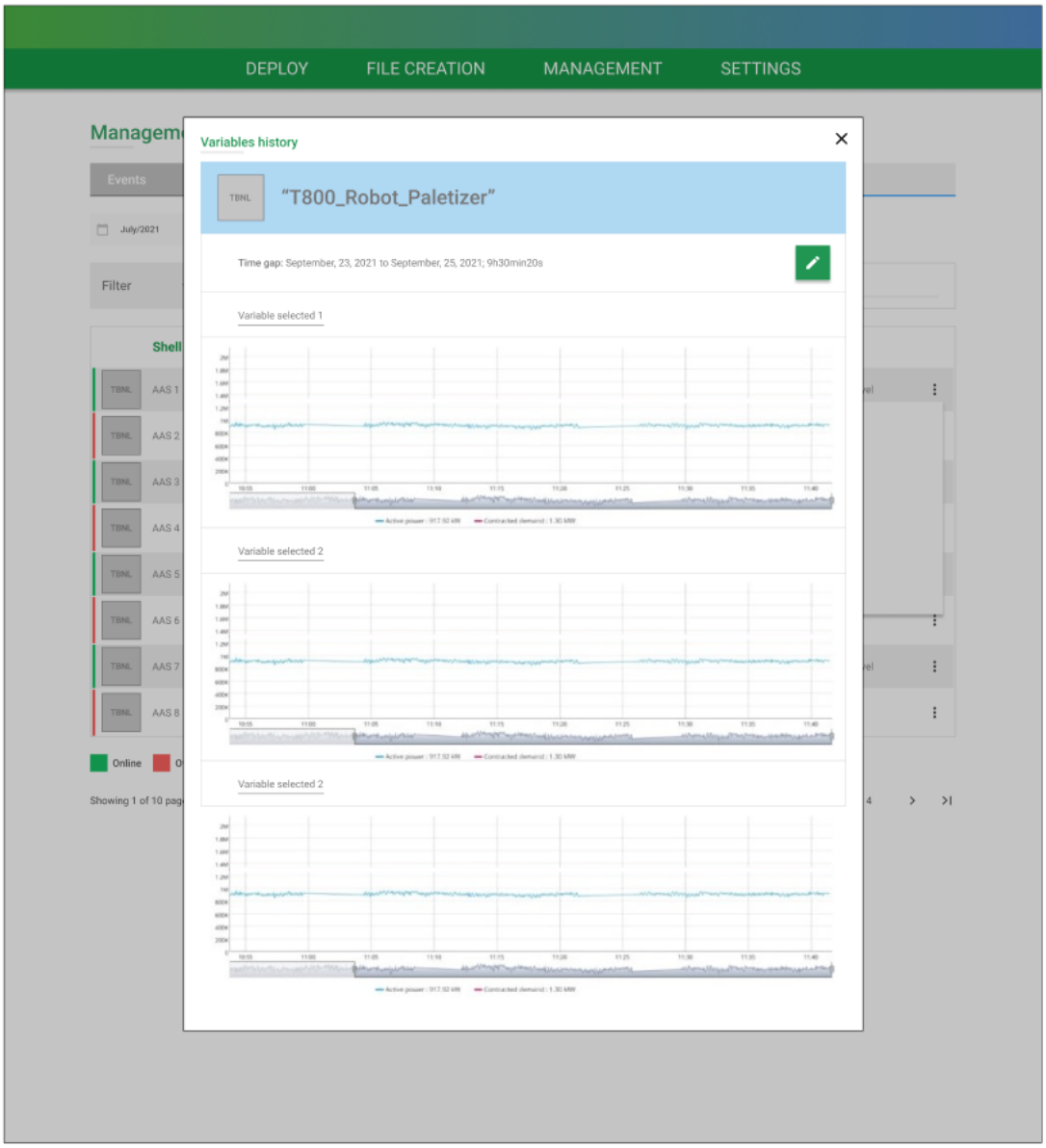



Fonte: (Elaborado pelo autor, 2023)

Após a seleção de variáveis um modal é aberto com os dados historizados como demonstrado na figura 54 abaixo. Com esses os dados é possível ajustar parâmetros e configurações de ativos em tempo real para otimizar processos industriais e maximizar a eficiência da produção.

A implementação de um histórico de variáveis em tempo real em um *AAS* proporciona uma vantagem significativa em ambientes industriais ao permitir respostas imediatas a eventos e condições operacionais em constante mudança. Isso contribui para a eficiência, segurança e confiabilidade das operações industriais.

Figura 54 – Modal de histórico de variáveis




Fonte: (Elaborado pelo autor, 2023)

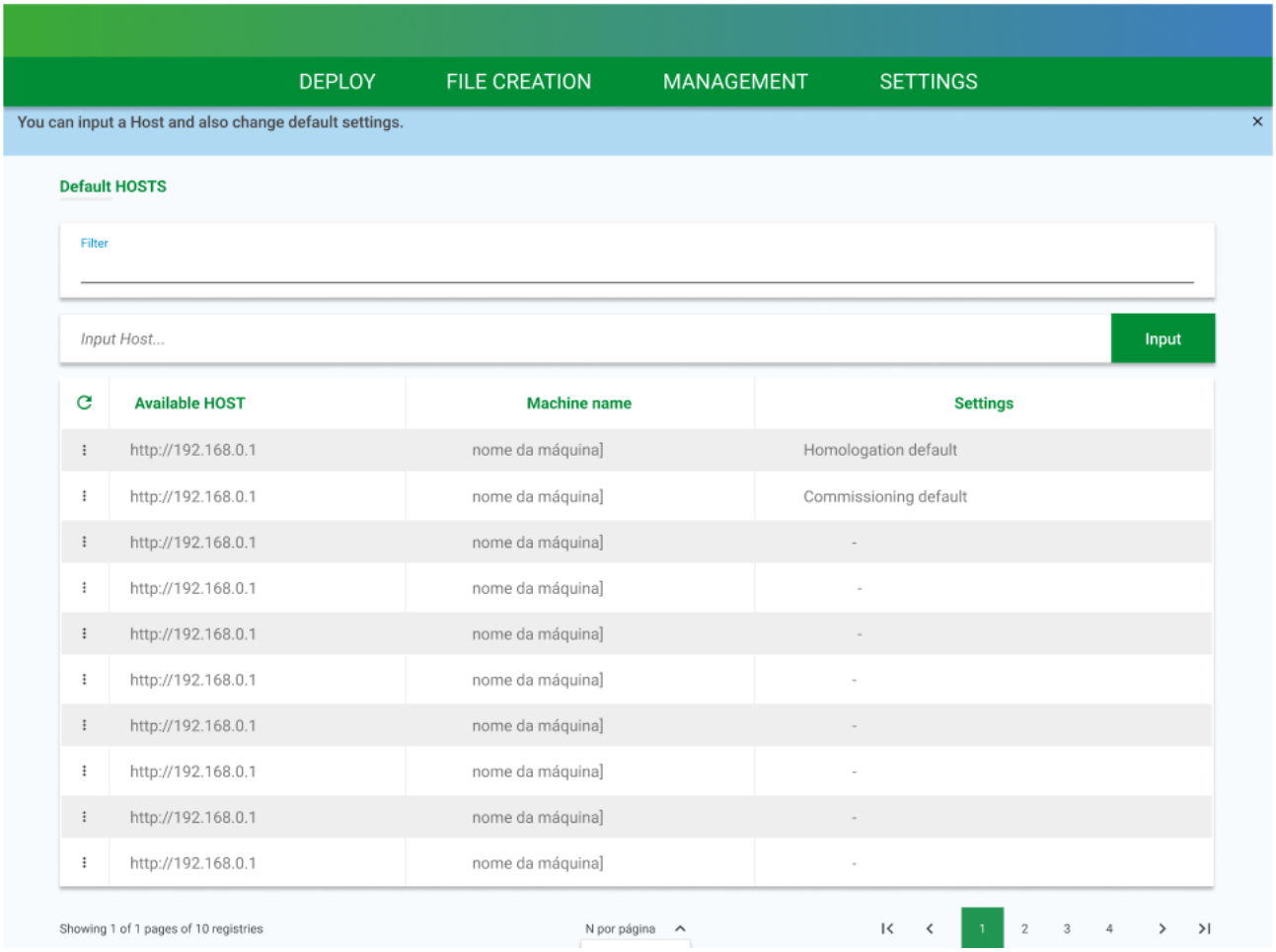
## 4.5 Configurações dos Shells

Nessa seção são abordadas as configurações dos *Asset Administration Shell*s, host padrão, *AAS* templates, submodelos básicos da empresa.

### 4.5.1 Hosts padrão

Nesta página operador pode realizar a configuração de hosts padrões de implantação que são utilizados no Passo 7 da implantação 4.3.1.8, os hosts configuráveis são de comissionamento e homologação como demonstrado abaixo na figura 55.

Figura 55 – Página de configuração de hosts



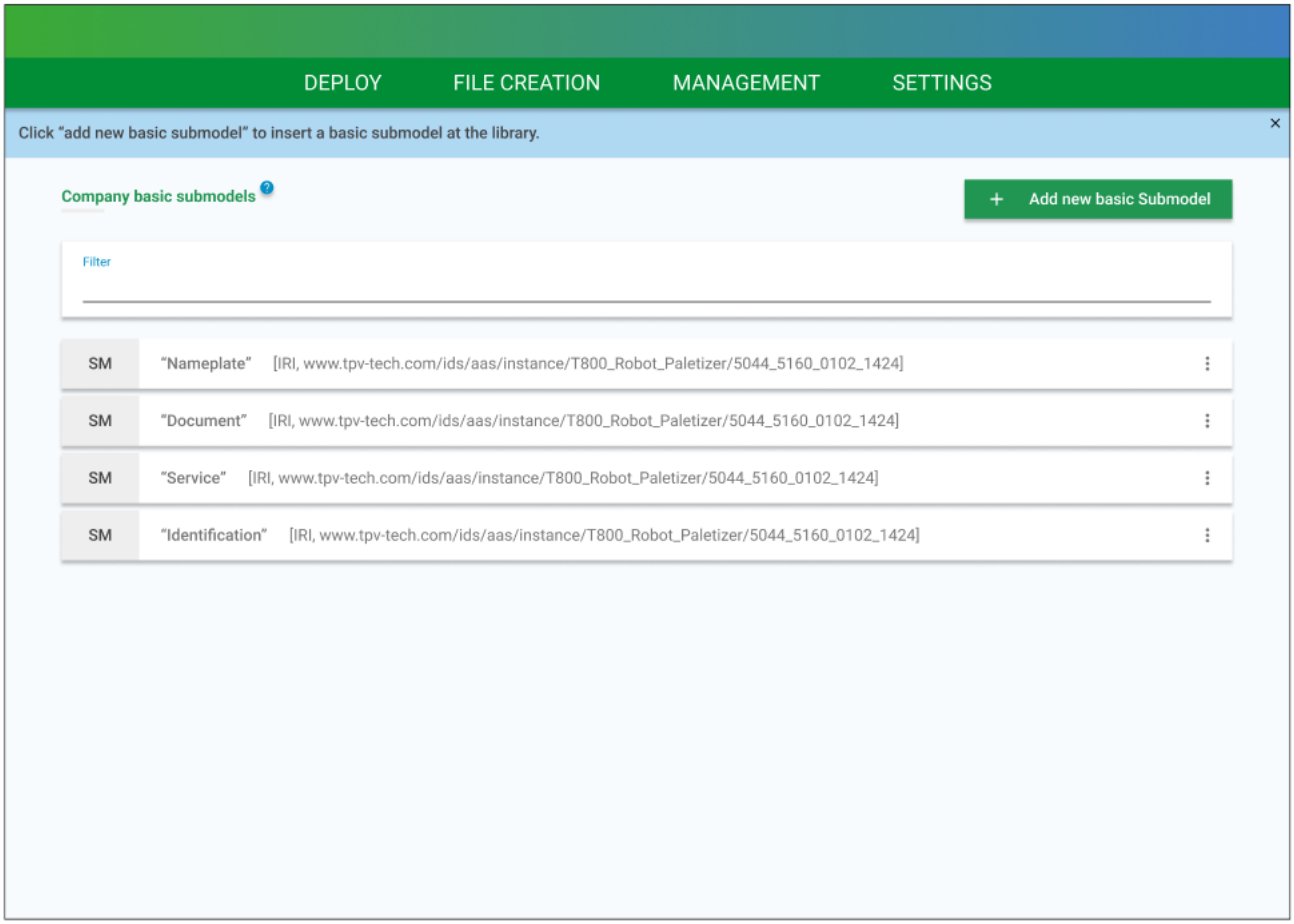

Fonte: (Elaborado pelo autor, 2023)

### 4.5.2 Submodelos básicos

Esta página possui a listagem contendo os submodelos básicos e padronizados da empresa como representado na figura 56· Caso algum submodelo apresente algum

problema de modelagem, poderá ser editado nesta página. Além disso, o operador pode inserir novos submodelos que podem ser importados no módulo de criação e modelagem de Shells.

Figura 56 – Página de submodelos básicos

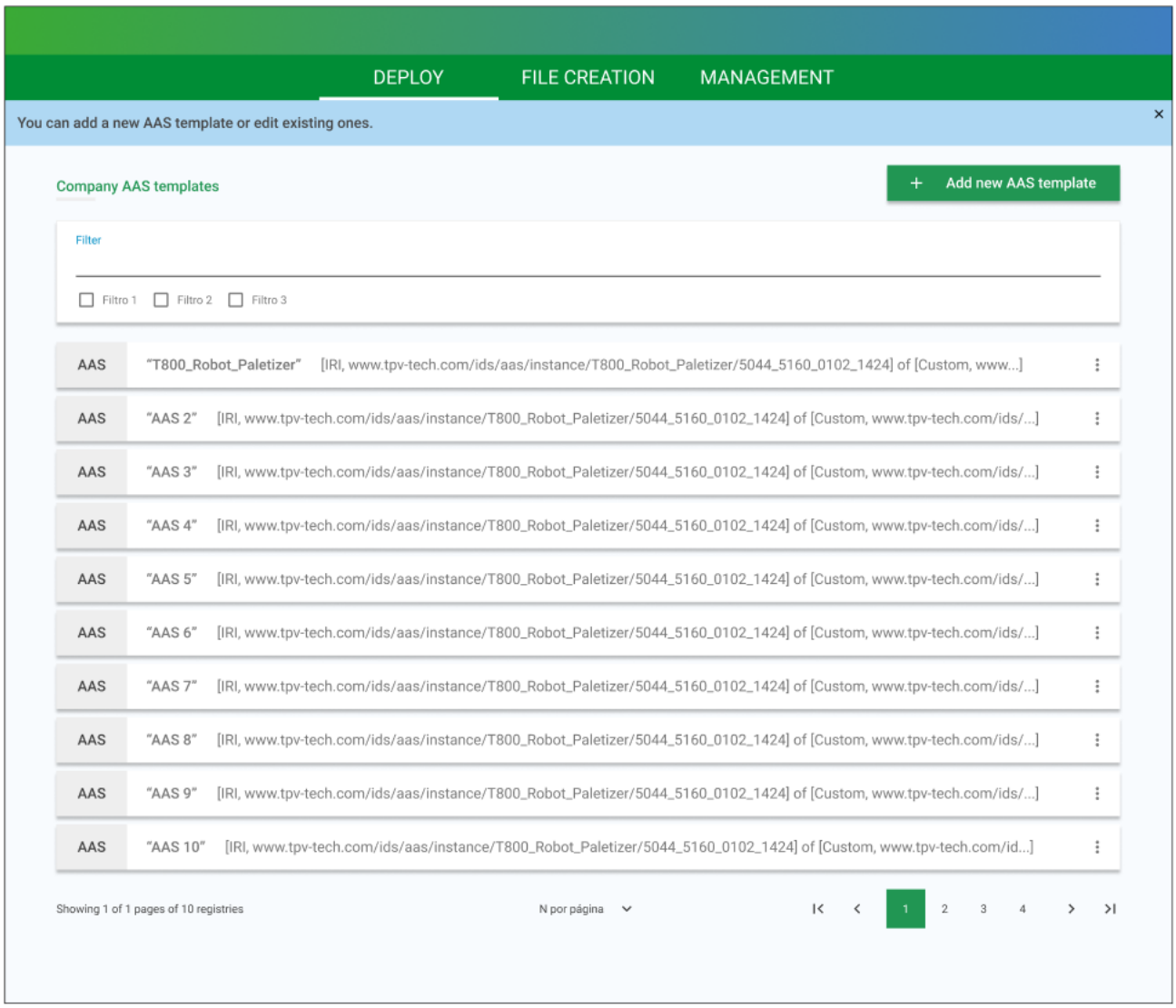



Fonte: (Elaborado pelo autor, 2023)

### 4.5.3 *AAS* templates

Esta página possui a listagem contendo os *Asset Administration Shell*s, como representado na figura 57· Esses *AAS*s podem ser padronizados por tipos de implantação, contendo os submodelos básicos que a empresa padronizou. Além disso, os templates ajudam na implantação de múltiplos gêmeos digitais em escala.

Figura 57 – Página de submodelos básicos




Fonte: (Elaborado pelo autor, 2023)

## 4.6 Testes

Nesta seção, serão apresentados os resultados dos testes do sistema de gerenciamento de Shells. A análise abrange os módulos de criação, implantação, gerenciamento, e configurações.

### 4.6.1 Estudo de caso

De forma geral os testes foram realizados e validados em ambiente fabril em uma planta do *Polo Industrial de Manaus* que realiza montagem de monitores e televisores. A planta possui gêmeos digitais criados com base em *Asset Administration Shell* para a representação virtual dos assets, alguns desses gêmeos digitais fazem monitoramentos de equipamentos vinculados a certificação da ISO 50001, onde é monitorado parâmetros elétricos e mecânicos. Além disso, esses dados coletados são utilizados para auditoria, manutenção e eficiência energética.

Os testes no sistema foram realizados por dois operadores que geravam as modelagem dos *Asset Administration Shell* de forma manual para que pudessem interagir da melhor forma possível com o sistema, com isso foi feita um bateria de testes desde o módulo de criação até a implantação e gerenciamento das representações digitais. Ambos os operadores eram empregados da planta. Além disso, a quantidade de testes não é padronizada.

Dessa forma os testes estão classificados como sucesso, falha e bloqueio onde:

- **Sucesso:** O status de sucesso é um indicador positivo de que o software atende aos requisitos especificados e funciona conforme esperado. Isso garante a qualidade e a confiabilidade do sistema.
- **Falha:** Se o teste resultou em falha, é abordado esse problema de maneira estruturada para garantir que os problemas identificados sejam corrigidos e que o software alcance os padrões de qualidade desejados.
- **Bloqueio:** Quando um teste é marcado como bloqueado, significa que houve algum impedimento a execução ou a conclusão bem-sucedida do teste.

Na tabela 24 abaixo, é apresentado a métrica total dos teste realizados durante o desenvolvimento do projeto. No total foram realizados 1.524 testes nos módulos implementados no sistema.

| **Status dos Testes** | **Quantidade** |
|---|---|
| Sucesso | 1203 |
| Falha | 291 |
| Bloqueado | 30 |
| Total | 1524 |

Tabela 24 – Testes realizados

Na tabela 25 abaixo, é apresentado as métricas dos teste divididas por módulos. Além disso, é possível observar que em todos os módulos do sistema, os números são maioritários em testes com status de SUCESSO. Com isso, é possível afirmar os módulos tem mais de 70% de sucesso nos testes.

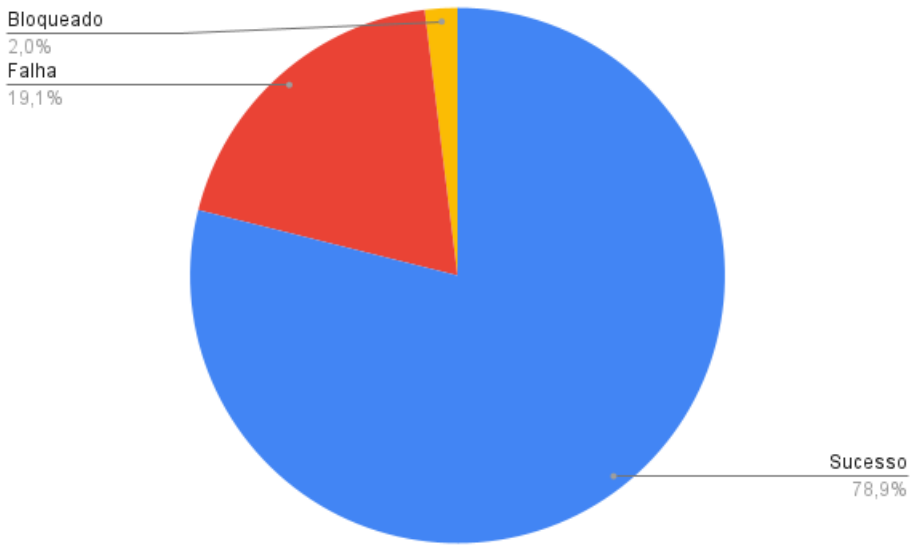
Tabela 25 – Testes por módulo

| **Módulo** | **Sucesso** | **Falha** | **Bloqueado** | **Quantidade** | **Sucesso %** |
|---|---|---|---|---|---|
| Gerenciamento | 270 | 75 | 8 | 335 | 80.60 % |
| Implantação | 336 | 131 | 10 | 477 | 70.44 % |
| Criação | 333 | 83 | 10 | 426 | 78.17 % |
| Configuração | 264 | 20 | 2 | 286 | 92.31 % |

Fonte: (Elaborado pelo autor, 2023)

Na figura 58, é apresentado um gráfico disposto em porcentagem com a totalidade dos testes. Com embasamento nos dados mostrados no gráfico de 78.9% de sucesso nos testes gerais, é possível afirmar que o sistema tem altas taxas de eficácia no gerenciamento considerando que é apenas um protótipo.

Figura 58 – Gráfico de testes




Fonte: (Elaborado pelo autor, 2023)

# 5

# Conclusão

O presente trabalho mergulhou nas complexidades do gerenciamento de *Asset Administration Shell* (*AAS*) em um contexto profundamente transformador da *Indústria 4.0*. Ao longo do projeto, foi possível entender a importância desta tecnologia na otimização da gestão de ativos digitais e na preparação das organizações para futuros desafios e oportunidades.

Através dos resultados obtidos pode-se perceber que a representação digital proporcionada pelos *AAS*s realmente garante a melhor gestão dos ativos. Se destacam, entre suas vantagens, a capacidade de os monitorar em tempo real, analisar dados preditivos e realizar manutenção proativa destes.

A padronização e interoperabilidade oferecidas pelos *AAS*s são partes cruciais para a integração bem-sucedida em ambientes industriais complexos. A capacidade de reunir informações de diferentes ativos e sistemas, independentemente do fornecedor, é um passo significativo em direção a uma colaboração mais eficaz e uma operação mais fluida. O sistema proposto atua eficazmente e de forma primordial a padronização e interoperabilidade de ativos.

Acerca dos resultados obtidos, foi possível concluir que o sistema desenvolvido neste projeto é capaz de atender as necessidades de criação, implantação, configuração e gestão de Shells centralizadas em uma só aplicação web. Vale ressaltar que o sistema elaborado neste projeto pode ser aplicado nas empresas do *Polo Industrial de Manaus* como solução inicial para gerir seus gêmeos digitais que utilizam como tecnologia base *Asset Administration Shell*s.

Agradeço a todos os envolvidos neste estudo, às organizações que abriram suas portas para a o projeto e aos colegas e mentores que apoiaram este empreendimento. Que esta pesquisa sirva como um farol para futuras inovações e avanços no gerenciamento de Shells no contexto da *Indústria 4.0*.

## 5.1 Sugestão de trabalhos futuros

Como sugestão de trabalhos futuros e melhorias do sistema recomenda-se os tópicos listados abaixos.

- **Segurança Cibernética em Ambientes de Shells:** Investigar e desenvolver estratégias avançadas de segurança cibernética para proteger os dados e informações contidos nos Shells.
- **Integração com Tecnologias de Internet das Coisas (IoT):** Explorar como os Shells podem ser integrados e otimizados em ambientes que fazem uso extensivo de sensores e dispositivos IoT.
- **Avaliação do Impacto Ambiental:** Investigar como o gerenciamento de Shells pode contribuir para práticas mais sustentáveis na indústria, reduzindo o impacto ambiental, como por exemplo, eficiência energética.
- **Desenvolvimento de Ferramentas de Monitoramento Avançadas:** Criar ferramentas mais avançadas e específicas para o monitoramento e gestão de Shells, integrando tecnologias emergentes como Inteligência Artificial e Aprendizado de Máquina.
- **Evolução dos Shells na Próxima Década:** Investigar tendências tecnológicas e como os Shells podem evoluir para atender às demandas futuras da indústria.

Essas sugestões oferecem uma gama de direções para pesquisas futuras. Cada uma delas representa uma oportunidade de aprofundar e expandir o conhecimento nessa área em constante evolução.

# REFERÊNCIAS


Amazon Web Services. *Usar WebSockets com distribuições do CloudFront*. 2023. Disponível em: <https://docs.aws.amazon.com/pt_br/AmazonCloudFront/latest/DeveloperGuide/distribution-working-with.websockets.html>. 53

Angular. *What is Angular?* 2022. Disponível em: <https://angular.io/guide/what-is-angular>. 25

HAAG, S.; ANDERL, R. Digital twin – proof of concept. *Manufacturing Letters*, v. 15, 02 2018. 20

HOBERMAN, S. *Data Modeling for MongoDB: Building Well-Designed and Supportable MongoDB Databases*. [S.l.]: Technics Publications, 2014. 29

IDTA. *AASX Package Explorer*. 2023. Disponível em: <https://github.com/admin-shell-io/aasx-package-explorer>. 52

JOHN S RINALDI. *OPC UA Cliente vs. Servidor*. 2018. Disponível em: <https://www.rtautomation.com/rtas-blog/opc-ua-client-vs-server/>. 53

MongoDB. *What is MongoDB?* 2023. Disponível em: <https://www.mongodb.com/docs/manual/>. 29

OLSEN, T. L.; TOMLIN, B. Industry 4.0: Opportunities and challenges for operations management. *Manufacturing & Service Operations Management*, INFORMS, v. 22, n. 1, p. 113–122, 2020. 19

OPC Foundation. *Unified Architecture*. 2023. Disponível em: <https://opcfoundation.org/about/opc-technologies/opc-ua/>. 53

Plattform Industrie 4.0. *Asset Administration Shell Reading Guide* . 2022. Rev. 3.9. Disponível em: <https://www.plattform-i40.de/IP/Redaktion/DE/Downloads/Publikation/AAS-ReadingGuide_202201.pdf?__blob=publicationFile&v=4>. 19, 20, 24

PLATTFORM INDUSTRIE 4.0. *Part 1 - The exchange of information between partners in the value chain of Industrie 4.0*. [S.l.], 2022. 3.0RC02. 24

Plattform Industrie 4.0. *Details of the Asset Administration Shell*. 2023. 3.0RC02. Disponível em: <https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/Details_of_the_Asset_Administration_Shell_Part1_V3.pdf?__blob=publicationFile&v=1>. 61, 62, 63, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78

Python Software Foundation. *What is Python? Executive Summary*. 2023. Disponível em: <https://www.python.org/doc/essays/blurb/>. 28, 29

RabbitMQ. *Consumers*. 2023. Disponível em: <https://www.rabbitmq.com/consumers.html>. 54

STOJANOVIC, L. et al. Methodology and tools for digital twin managementmdash;the fa3st approach. *IoT*, v. 2, n. 4, p. 717–740, 2021. ISSN 2624-831X. Disponível em: <https://www.mdpi.com/2624-831X/2/4/36>. 20

TypeScript. *TypeScript é JavaScript com sintaxe para tipos*. 2022. Disponível em: <https://www.typescriptlang.org/>. 25

Unified Automation. *UaExpert—A Full-Featured OPC UA Client*. 2023. Disponível em: <https://www.unified-automation.com/products/development-tools/uaexpert.html>. 52