CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA
Faculdade de Tecnologia de Mauá
Curso Superior de Tecnologia em Desenvolvimento de Software Multiplataforma

Augusto De Barros Almeida
Bruno Henrique Gusmão Gonçalves
Renato Fernandes Da Silva
Rodrigo Bastos Amaral

S.A.O.R.I – SISTEMA AVANÇADO DE ORGANIZAÇÃO RECURSOS E INSUMOS

Mauá
2025

AUGUSTO DE BARROS ALMEIDA
BRUNO HENRIQUE GUSMÃO GONÇALVES
RENATO FERNANDES DA SILVA
RODRIGO BASTOS AMARAL

S.A.O.R.I – SISTEMA AVANÇADO DE ORGANIZAÇÃO RECURSOS E INSUMOS

Relatório técnico-científico apresentado à Fatec de Mauá, como exigência parcial para obtenção do título de Tecnólogo em Desenvolvimento de Software Multiplataforma, sob a orientação da profa. Me. Luana Lourenço.

Mauá
2025

AUGUSTO DE BARROS ALMEIDA
BRUNO HENRIQUE GUSMÃO GONÇALVES
RENATO FERNANDES DA SILVA
RODRIGO BASTOS AMARAL

S.A.O.R.I – SISTEMA AVANÇADO DE ORGANIZAÇÃO RECURSOS E INSUMOS

Trabalho de Conclusão de Curso apresentado à Faculdade de Tecnologia de Mauá como requisito parcial para a obtenção do título de Tecnólogo (a) em Desenvolvimento de Software Multiplataforma

Trabalho de Conclusão de Curso apresentado e aprovado em: xx/12/2025 Banca Examinadora:

Orientador: Prof. Me. Luana Lourenço – Orientador

Prof. Ivan Carlos Pavão, FATEC Mauá– Avaliador

Prof. Carlos Ronny De Sousa, FATEC Mauá – Avaliador

Esse trabalho é dedicado aos nossos pais e esposas, e a todos os professores, aqueles que me ajudaram e aos colegas que de alguma forma contribuíram para a conclusão deste trabalho. “O sucesso é ir de fracasso em fracasso sem perder o entusiasmo”.

WINSTON CHURCHILL

RESUMO

Este trabalho visou criar um sistema de geração de ordens de materiais de maneira simples e funcional, facilitando o trabalho do técnico de campo para evitar pedidos de materiais incorretos ou incompletos, evitando assim perda de tempo e atrasos na entrega de projetos devido a materiais errados ou dificuldades para o setor de compras na hora de determinar exatamente o que é necessário.

Ele é dotado de uma parte feito em *Desktop* em Python para gerar o pedido de materiais, escolhendo por meio filtro de requisitos exatamente o que procura, e uma segunda parte do *software* que funciona em *web* para fazer a parte de gestão do mesmo, onde as liberações do pedido pela parte gerencial serão executadas e assim vai para até o departamento de compras.

Funciona como uma parte de um sistema MRP, mas para facilitar o papel do funcionário que determina que peças usar, sem a necessidade do mesmo olhar diversos catálogos de peças, de múltiplos fornecedores, que além de extensos, são extremamente confusos. Facilitando assim a obtenção de uma ordem de compra de maneira rápida e assertiva.

Palavras-chave: Python, MRP, BOM, Lista de peças, Sistema de pedidos

ABSTRACT

This work aimed to create a system for generating material orders in a simple and functional way, making the field technician’s job easier by preventing incorrect or incomplete material requests. This helps avoid wasted time and project delivery delays caused by wrong materials or difficulties in the purchasing department when determining exactly what is needed.

The solution consists of two parts: a desktop application, developed in Python, which generates the material order by selecting exactly what is needed through a requirements filter; and a web-based module, which handles the management side, where managerial approvals of the order are carried out before it is forwarded to the purchasing department.

It operates as a component of an MRP system, but is designed to simplify the role of the employee responsible for selecting the required parts—eliminating the need to consult numerous and often confusing supplier catalogs. This makes it possible to obtain a purchase order quickly and accurately.

Keywords: Python, MRP, BOM, Parts List, Ordering System

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

B.O.M. – *Bill of Materials* / Lista de Materiais

Desktop – Computador de mesa

Feature – Característica

Lean Manufacturing – Produção Enxuta

MRP – *Material Requirements Planning* / Planejamento de Necessidades de Materiais

S.A.O.R.I. – Sistema Avançado de Organização de Recursos e Insumos

SCRUM – desenvolvimento ágil de software

SWOT – Strenghts (Forças), Weaknesses (Fraquezas), Opportunities (Oportunidades) e Threats (Ameaças).

Workflow – Fluxo de trabalho

SUMÁRIO

1 INTRODUÇÃO

Com o aumento da tecnologia e com novas abordagens do que é necessário para se produzir, novas maneiras e métodos de produção focados em redução de custos ao produto final cada vez mais é necessário mitigar erros durante as fases de um projeto.

Um pedido errado antigamente poderia não significar muito, mas projetos complexos que exigem que uma fase seja executada para que outra possa seguir em frente, demanda que um erro atrasa toda uma cadeia de eventos. Uma peça errada devido a um pequeno erro no código hoje não é mais tolerável.

Para isso os sistemas informatizados ajudam e muito, mas mesmo assim, precisa que o profissional saiba exatamente o código do produto que deseja solicitar, muitas vezes perde-se um tempo precioso na correta obtenção do código do produto para assim lançar o mesmo no sistema, produtos de função similar tem códigos diferentes o que demanda na falta do mesmo no mercado um novo tempo de pesquisa.

O Sistema S.A.O.R.I. vêm para facilitar essas duas fases do processo sendo que no mesmo código estará o código de diversos fornecedores no caso de produtos semelhante e também estará presente no programa uma lista de filtragem que conforme é preenchida mostra as opções de produtos mais adequadas, sem a necessidade de acessar nenhum catálogo de fornecedor, poupando tempo e agilizando o processo complicado e trabalhoso de se obter uma lista de peças.

1.1 Fundamentação teórica

Apresenta uma abordagem das teorias e conceitos de fontes fidedignas que dão sustentação ao desenvolvimento do projeto.

1.2 Metodologia

Mostra métodos e técnicas para percorrer o caminho da pesquisa e construção lógica do projeto.

1.3 Desenvolvimento do projeto

Encontra-se o passo a passo da construção do projeto. Intitula-se S.A.O.R.I – Sistema Avançado de Organização Recursos e Insumos

1.4 Acessibilidade

Descritivo detalhado de tecnologias e funcionalidades presentes e/ou à serem implementadas para garantir amplo acesso ao sistema.

1.5 Considerações finais

Destacam-se os objetivos propostos com justificativas, vantagens e desvantagens, pontos fortes e fracos e possíveis sugestões para futuros trabalhos.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são abordadas teorias que dão sustentação à necessidade e explicação do Sistema Avançado de Organização Recursos e Insumos.

2.1 Proteção da Produção

As organizações modernas enfrentam ambientes turbulentos, onde mudanças rápidas podem afetar a produção. Para minimizar impactos externos, é importante proteger a produção, entre elas as duas formas principais são: proteção física e proteção organizacional.

Proteção física envolve manter estoques de recursos e produtos acabados para que interrupções no fornecimento ou na demanda não interrompam a operação. A ideia é ter “estoques de proteção”, que garantam que a produção continue mesmo em situações imprevistas, como falta de insumos ou picos de demanda.

A proteção organizacional refere-se à alocação de responsabilidades e recursos dentro da empresa. Por exemplo, garantir que pessoas, tecnologia e materiais essenciais à produção estejam protegidos de incertezas externas. Essa proteção cria uma barreira contra problemas inesperados e permite que a empresa funcione de forma mais estável, eficiente e resiliente.

Ambas as proteções geram problemas, pois além de deixar o negócio não preparado as oscilações do mercado, geram uma ótima oportunidade de desculpas por parte dos gerentes de produção, estoques altos tem custos elevados e quando envolve o cliente esperar por serviços pode gerar insatisfações. (SLACK, CHAMBERS, JOHNSTON, 2002)

2.2 Lean Manufacturing

Dentro da logística é muito importante que os processos sejam os mais enxutos possíveis, pois isso reduz custos e aumenta muito a capacidade de concorrência, um sistema de gestão focado na redução do desperdício, tem um ganho também na

qualidade. Dentre os mais comuns métodos de redução de desperdício temos o *Lean Manufacturing* que tem como objetivo central entregar o máximo de valor com a menor quantidade de recursos possíveis.

Entre as principais vantagens do *Lean Manufacturing* está a organização dos processos, que deixa o chão de fábrica mais limpo e arrumado, com estoques menores liberando mais espaço físico, qualidade aumenta, pois há menos retrabalho e maior produtividade de máquinas e operadores, a produção ajusta os recursos à demanda, evitando ociosidade e custos extras, com isso, os materiais são transformados em menos tempo e com menos recursos, enquanto os operadores participam mais e se tornam parte estratégica da produção. (SENAI ES, 2018)

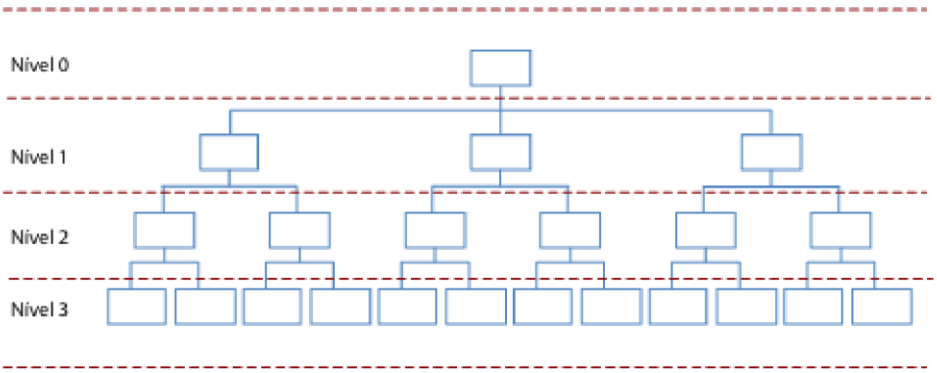
2.3 Estrutura de Produto

A estrutura de produto, também conhecida como árvore do produto, é um diagrama que mostra os componentes de um produto e a ordem pela qual as peças componentes são agrupadas. Representa uma estrutura de produto que mostra todas as relações pai-filho entre seus itens e informa a composição do produto, mostra como cada componente se agrupa para formar o item final, sendo que o nível 3 são os itens básicos e nível 0 é o produto acabado.

Para facilitar, muitas empresas usam a lista de materiais “indentada”, que organiza os componentes em níveis com suas quantidades.

O nível 0 é o produto acabado, os níveis seguintes representam subconjuntos e, por último, os itens básicos. Para facilitar, muitas empresas usam a lista de materiais “indentada”, que organiza os componentes em níveis com suas quantidades.

Figura 1 - Representação de Lista de materiais como uma estrutura de produto



GRAZIANI, 2021, p.214

Algumas peças podem servir a vários itens de origem, prática chamada padronização ou modularização, aumentando volume de produção, reduzindo custos e estoques. Essa estrutura permite calcular as necessidades brutas: saber exatamente o que e quanto comprar ou produzir para atender à necessidade/demanda. (GRAZIANI, 2021)

2.3 Lista de materiais (B.O.M.)

Segundo (TOLEDO, 1996) lista de materiais é um dos principais elementos para a integração dos sistemas de manufatura, porque ela flui por quase todos os departamentos de uma empresa.

Apesar da pouca importância que é atribuída a lista de materiais. A lista de materiais constitui a base do sistema de informação usado na gestão da produção e no controle do inventário.

A lista de materiais transforma a representação gráfica da estrutura multinível do produto numa representação linear dos diversos relacionamentos existentes entre matéria-prima, componentes, submontagens, montagens e produto final. Além disso, uma lista de materiais bem estruturada garante maior precisão no planejamento das necessidades de materiais, reduzindo retrabalhos e falhas no processo produtivo.

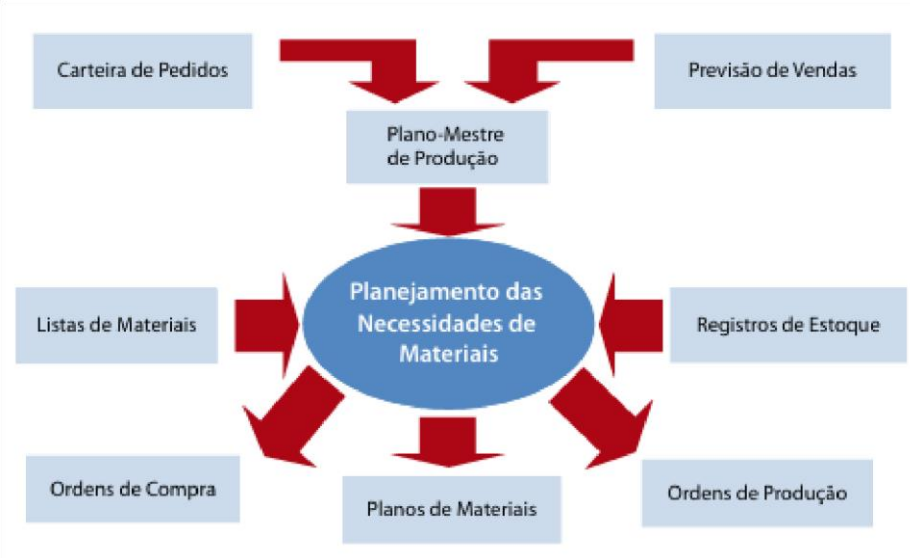
2.4 MRP

Segundo (GRAZIANI, 2021) MRP é um sistema computacional que a partir dos dados de demanda futura e pedidos em carteira, calcula as necessidades futuras de materiais para atendimento da demanda, sem que ocorram atrasos e nas quantidades corretas, sendo que basicamente todos os processos de produção dependem da compra de matérias primas.

O cálculo de necessidades brutas, realizado a partir das informações da lista de materiais, responde a duas questões fundamentais: o que produzir e comprar; e quanto.

A questão pertinente é saber quando efetuar as ações gerenciais de comprar ou produzir. Para respondê-la, precisamos primeiramente conhecer os tempos de obtenção dos diversos itens (*lead time*), sejam eles comprados ou produzidos. Pela lógica utilizada no ambiente MRP, o tempo de obtenção é o tempo que decorre entre a liberação de uma ordem e o material correspondente estar pronto e disponível para uso. Comprar matérias primas o mais tarde possível diminui os estoques e quebras no armazenamento, diminuindo perdas, que somente onera o produto acabado.

Figura 2 - Esquema do planejamento de necessidades de materiais



GRAZIANI, 2021, p.218

2.5 Workflow

Hoje em dia, o trabalho em equipe é essencial para empresas que buscam qualidade e agilidade, mas muitas vezes o sucesso depende de boa comunicação e integração entre os membros. Métodos tradicionais como circulação de papéis, memorandos, telefonemas ou quadros de aviso geram problemas como excesso de papel, informações inconsistentes, reuniões improdutivas e comunicação ineficiente.

Com o avanço da tecnologia da informação nos últimos anos, o trabalho colaborativo surge como uma solução para superar os inconvenientes dos métodos tradicionais e oferece novos recursos para o trabalho em grupo, permitindo que pessoas separadas fisicamente colaborem, auxiliando na tomada de decisão, fortalecendo a sinergia entre os membros das equipes, melhorando a produção de documentos, projetos e especificações, além de manter todos informados sobre eventos importantes.

O *workflow* surge como uma solução tecnológica para esses desafios. Ele permite que pessoas fisicamente separadas trabalhem juntas de forma coordenada, auxilia na tomada de decisão, fortalece a sinergia da equipe e melhora a produção de documentos, projetos e especificações. Além disso, mantém todos informados sobre eventos importantes e garante consistência e segurança no controle da informação, facilitando a gestão de processos dentro da organização. (PITHON, 2004)

3 METODOLOGIA

Neste capítulo encontra-se a trajetória para o desenvolvimento e construção do projeto. Trata-se de uma pesquisa aplicada que é desenvolvida em home office por nós integrantes do grupo desenvolvimento.

Segundo Prodanov e Freitas (2013), a metodologia consiste em estudar, compreender e avaliar os métodos disponíveis para realização de uma pesquisa

acadêmica, tornando necessário a aplicação de técnicas para a construção do conhecimento. Enfatiza que os métodos são procedimentos amplos do raciocínio, enquanto as técnicas são procedimentos mais restritos que operacionalizam os métodos mediante emprego de instrumentos adequados

Dentre os vários autores de metodologia científica, Severino (2013) aponta que a metodologia é a preparação metódica e planejada de um trabalho cientifico que se encontra inserida em uma sequência de etapas para sua construção, compreendendo o tema-problema e justificativa, levantamento bibliográfico, seleção da bibliografia, desenvolvimento da construção do projeto e redação do texto.

3.1 Tema-problema com justificativa e descrição funcional do projeto

O tema abordado Sistema Avançado de Organização de Recursos e Insumos, foi escolhido devido a ser uma *feature* não disponível atualmente em sistemas MRP, apesar dela facilitar imensamente o trabalho do desenvolvedor.

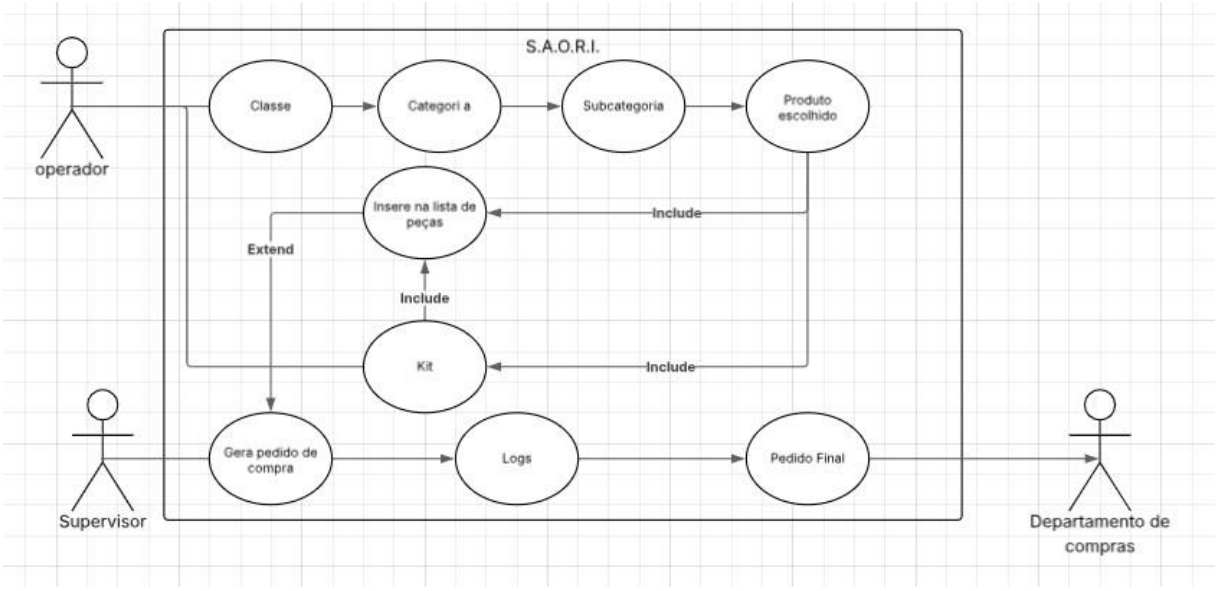
Trata-se de um sistema de criação de lista de materiais feito em formato *desktop* com capacidade de pesquisa interna, tirando a necessidade de pesquisar em imensos catálogos de peças, sendo que peças que tem mais de um fornecedor cadastrado, tem o mesmo código, facilitando a troca do mesmo no caso de falta no mercado ou elevado *lead time* para a compra da peça. Durante a fase de escolha o operador escolhe um fornecedor específico, mas uma pesquisa rápida e ele pode trocar facilmente.

O sistema também permite a criação de *kits* de peças, que nada mais é que um conjunto de peças comuns, para em compras futuras facilite e não precise pesquisar os itens um a um, facilitando assim compras futuras e novas listas. Imagine peças para montar um painel e que várias máquinas usem o mesmo painel, basta que ele crie um kit chamado painel (nome ruim usado apenas como exemplo, por ser pouco descritivo), toda vez que for fazer uma máquina basta inserir esse kit chamado painel que todas as peças do mesmo são inseridas automaticamente.

Depois da criação da lista de peças o sistema *web* exibe a lista e busca no banco de dados uma lista de superiores que precisam aprovar a lista antes de ela ser

passada para o departamento de compras. Cada superior pode liberar ou reprovar a lista de peças, integralmente ou parcialmente. O sistema gera logs avisando quando foi aprovada a lista e por quem, como também gera logs de quais peças foram reprovadas e por quem, para posterior auditoria do processo.

Figura 3 – Diagrama de caso de uso



Fonte: Autoria própria, 2025

3.2 Etapas teóricas e práticas para o desenvolvimento do software

Após delimitar o tema-problema, justificativa e descrição do projeto parte-se para as seguintes etapas:

Primeira etapa: reunião dos integrantes do grupo para traçar as diretrizes de como efetuar as pesquisas e andamentos dos trabalhos. Usamos algumas ferramentas da qualidade e gestão de projetos, tais como: SWOT, SCRUM. Marcou um dia da semana para apresentar o andamento dos trabalhos. Inicialmente, iniciou-se com a análise SWOT ou FOFA (Strengths - Forças), Weaknesses - Fraquezas, Opportunities - Oportunidades and Threats - Ameaças). A Figura 2.2 mostra a análise SWOT.

Figura 4 - Análise SOWT

Fonte: Autoria própria, 2025

A Figura 2.2 mostra os principais objetivos para consagrar a ideia do projeto, fazendo uma análise SWOT[1] que segundo NETO (2011 apud COSTA,2001, p.16) é usado para, “estimar pontos fortes e fracos, oportunidades e ameaças, a fim de desenvolver planos de médio e longo prazo”.

---

[1] Montamos uma tabela SCRUM no Trello e foi combinado reuniões todas as terças feiras para atualizações e manutenções no código, afim de melhorar e organizar as tarefas entre os membros do grupo.

Segunda etapa: Uma vez consagrada a ideia central do projeto, delimitam-se os objetivos específicos e a partir dos mesmos, inicia-se o levantamento bibliográfico realizado na Fatec Mauá. Foram realizadas pesquisas em sites especializados, artigos e livros técnicos.

Terceira etapa: após o levantamento bibliográfico a seleção daqueles que possuíam maior aderência ao tema proposto para construir o Capitulo 1 – Fundamentação teórica.

Quarta etapa: estudo de quais ferramentas serão utilizadas para o desenvolvimento do *software*, foi escolhido para a parte *desktop* foi escolhido o Python devido a sua facilidade de implementação, muitas bibliotecas e interface facilitada usando o *customtkinter* que tem uma interface moderna. Para o desenvolvimento web será usado como ferramenta o *Flask.*

Quinta etapa: criação de testes automatizados para validar o software e evitar travamentos inesperados.

Sexta etapa: criação das telas de *login* e principal do aplicativo *desktop* com integração ao banco de dados *mysql* comunicação do banco de dados usando o *SqlAlchemy* para uma integração mais segura com o banco de dados impedindo problemas como o *injection.*

Sétima etapa: confecção de uma API feita com *FastApi* para comunicação com o aplicativo web, para que o mesmo possa se comunicar sem problema com o banco de dados. Cuidados extras para que a API tenha alta disponibilidade.

Oitava etapa: concluído o projeto elaborou-se a documentação final, as Considerações finais e Resumo.

3.3 Ficha técnica

Especificação	Detalhamento
Nome do sistema/software	S.A.O.R.I – Sistema Avançado de Organização de Recursos e Insumos

Versão	1.0 (versão inicial acadêmica)
Desenvolvedores	Augusto De Barros Almeida; Bruno Henrique Gusmão Gonçalves; Renato Fernandes da Silva; Rodrigo Bastos Amaral
Data de criação	2025
Plataformas	Aplicação desktop (Windows/Linux com Python 3); aplicação web (navegadores modernos com HTML5/ES6) [2]
Linguagens de programação	Python 3 (Desktop e API); JavaScript/TypeScript (Frontend web – branch web) [2][1]
Frameworks e bibliotecas	Tkinter/CustomTkinter (GUI desktop); Flask (frontend web); FastAPI (API REST); SQLAlchemy (ORM) [2][1]
Banco de dados	MySQL (principal), com suporte a PostgreSQL via abstração do SQLAlchemy
Requisitos de hardware	CPU 2.0 GHz ou superior; 4 GB de RAM; ~500 MB de espaço em disco
Requisitos de software	Python 3.8+; servidor MySQL; navegador compatível com HTML5
Ferramentas de desenvolvimento	Git/GitHub; Visual Studio Code; Trello (SCRUM);
Módulos principais	Módulo desktop de criação de listas de materiais e kits; módulo web de gestão/aprovação; API para integração com o banco de dados [2]
Funcionalidades principais	Geração de listas de materiais com filtros; criação de kits; múltiplos fornecedores por código; workflow de aprovação com logs e auditoria; apoio a MRP/BOM [2]
Recursos de segurança	Credenciais em variáveis de ambiente; prevenção de SQL Injection via SQLAlchemy; transações ACID; logs de auditoria [2]
Padrões de projeto	Factory, Repository, Singleton e Context Manager aplicados na camada de dados [2]

Objetivo	Reduzir erros e tempo na geração de ordens de materiais, integrando filtragem de peças, kits e fluxo de aprovação a processos de MRP [2]

4 DESENVOLVIMENTO DO PROJETO

No desenvolvimento do MyApp, a camada de acesso a dados representa um componente crítico para garantir a segurança, performance e confiabilidade do sistema. A escolha pelo SQLAlchemy como ORM (Object-Relational Mapping) principal foi fundamentada em sua maturidade, ampla adoção na comunidade Python e capacidade de abstrair as complexidades inerentes ao acesso a bancos de dados relacionais.

Esta seção detalha a arquitetura implementada para o acesso a dados, abordando desde a configuração da conexão até padrões avançados de projeto aplicados para garantir a facilidade na manutenção e escalabilidade do sistema.

4.1 Objetivo da Implementação

A implementação do módulo de acesso a dados teve como objetivos principais:

- **Abstração de Banco de Dados:** Fornecer uma interface unificada para diferentes SGBDs (PostgreSQL e MySQL)
- **Segurança:** Garantir o tratamento seguro de credenciais e operações de banco
- **Gerenciamento de Transações:** Implementar controle robusto de transações ACID
- **Manutenibilidade:** Facilitar a evolução do código através de padrões estabelecidos ● **Testabilidade:** Permitir a criação de testes unitários e de integração

4.2 Metodologia do Desenvolvimento

O desenvolvimento seguiu uma abordagem baseada em padrões de projeto (Design Patterns), com ênfase em:

- **Factory Pattern:** Para criação centralizada de sessões
- **Repository Pattern:** Para abstração das operações de persistência
- **Singleton Pattern:** Para garantir uma única instância de conexão
- **Context Manager Pattern:** Para gerenciamento automático de recursos

Esta implementação contribui com:

- **Modelo Replicável:** Arquitetura que pode ser reutilizada em outros projetos

- **Boas Práticas:** Exemplificação de padrões consagrados na comunidade Python
- **Documentação Técnica:** Descrição detalhada das decisões de implementação ● **Base para Evolução:** Fundamentos sólidos para futuras melhorias

4.3 Introdução ao SQLAlchemy

SQLAlchemy é uma biblioteca ORM (Object-Relational Mapping) para Python que facilita a interação com bancos de dados relacionais. Ela permite mapear objetos Python para tabelas de banco de dados, abstraindo operações SQL complexas. Neste projeto, SQLAlchemy foi escolhido por sua flexibilidade, suporte a múltiplos bancos de dados e integração nativa com Python, proporcionando um desenvolvimento mais produtivo e menos propenso a erros.

4.4 Análise Detalhada do Código

O código em app/database.py e factory.py configura a conexão com o banco de dados usando SQLAlchemy. Vamos analisar cada parte

4.5 Configurações do Banco

As configurações são carregadas de variáveis de ambiente usando a biblioteca dotenv:

Figura 5 - configuração do dotenv

```
load_dotenv()
USER = os.getenv("DB_USER")
PASSWORD = quote_plus(os.getenv("DB_PASSWORD"))
HOST = os.getenv("DB_HOST")
DB = os.getenv("DB_NAME")
```

Fonte: Autoria própria, 2025

Isso garante segurança, evitando hardcode de credenciais. A função (quote_plus codifica caracteres especiais na senha para URLs.

4.6 Factory Pattern para sessões

Implementamos um padrão Factory para gerenciar sessões do banco:

Figura 6 - Factory do banco de dados

```python
class SessionFactory:
    """Factory Singleton para Engine + Session SQLAlchemy."""

    _instance = None
    _engine = None
    _Session = None

    def __new__(cls):
        if cls._instance is None:
            cls._instance = super(SessionFactory, cls).__new__(cls)
            cls._instance._configure()
        return cls._instance
```

Fonte: Autoria Própria, 2025

4.7 Construção da URL do Banco

A URL de conexão é montada dinamicamente baseada no tipo de banco

Figura 7 - Construtor da URL de conexão

```python
def _configure(self):
    DatabaseConfigValidator.validate()

    db_type = os.getenv("DB_TYPE").lower()
    db_user = os.getenv("DB_USER")
    db_pass = os.getenv("DB_PASS")
    db_host = os.getenv("DB_HOST")
    db_port = os.getenv("DB_PORT")
    db_name = os.getenv("DB_NAME")

    if db_type == "postgres":
        url = f"postgresql+psycopg2://{db_user}:{db_pass}@{db_host}:{db_port}/{db_name}"
    elif db_type == "mysql":
        url = f"mysql+mysqlconnector://{db_user}:{db_pass}@{db_host}:{db_port}/{db_name}"
```

Fonte: Autoria própria, 2025

4.8 Configuração

Implementamos validação robusta das configurações:

Figura 8 - Configuração de validação

```python
class DatabaseConfigValidator:
    """Valida se variáveis obrigatórias estão presentes antes de iniciar."""

    REQUIRED_VARS = ["DB_TYPE", "DB_USER", "DB_PASS", "DB_HOST", "DB_PORT", "DB_NAME"]

    @classmethod
    def validate(cls):
        missing = [var for var in cls.REQUIRED_VARS if not os.getenv(var)]
        if missing:
            raise DatabaseConfigurationError(
                f"Variáveis de ambiente ausentes: {', '.join(missing)}"
            )
```

Fonte: Autoria própria, 2025

4.9 Gerenciamento de Sessões com Context Manager

Utilizamos um context manager para gerenciamento de transições de telas:

Figura 9 - Gerenciador de sessão

```python
@contextmanager
def get_session(self):
    """Gerencia sessão com commit/rollback automático."""
    session = self._Session()
    try:
        yield session
        session.commit()
    except Exception as e:
        session.rollback()
        logger.error(f"Erro na sessão: {e}")
        raise
    finally:
        session.close()
```

Fonte: Autoria própria, 2025

4.10 Padrão Repository

Implementamos o padrão Repository para abstrair o acesso a dados:

Figura 10 - Interface de Repository implementada para Auth

```python
class BaseRepository:
    """Classe base para repositórios."""

    def __init__(self, session):
        logger.debug(f"Inicializando BaseRepository com sessão: {session}")
        self.session = session

class AuthRepository(BaseRepository):
    """Repositório de autenticação."""

    def verify_credentials(self, username, password):
        logger.debug(f"Verificando credenciais para usuário: {username}")
        # Implementação de verificação
        return username == "admin" and password == "admin"
```

Fonte: Autoria própria, 2025

4.11 Aspectos Técnicos e Funcionamento

I. Segurança

- Credenciais em variáveis de ambiente
- Codificação de caracteres especiais
- Validação de configurações antes da inicialização

II. Padrões de Projeto Aplicados

- Singleton: Garante uma única instância da factory
- Factory Method: Cria sessões de forma centralizada
- Repository: Abstrai o acesso a dados
- Context Manager: Gerencia ciclo de vida das sessões

III. Fluxo de Operação

- Validação das configurações
- Criação do engine SQLAlchemy
- Configuração do sessionmaker
- Obtenção de sessões via context manager
- Execução de operações com commit/rollback automático

4.12 Testes Implementados

Desenvolvemos testes abrangentes para validar o funcionamento:

Figura 11 - Testes do gerenciador de sessões e configuração de validação

```python
def test_context_manager(monkeypatch):
    reset_factory_singleton()
    setup_env(monkeypatch)
    factory = SessionFactory()

    with factory.get_session() as session:
        assert isinstance(session, DummySession)

def test_configuracao_incompleta(monkeypatch):
    reset_factory_singleton()
    monkeypatch.delenv("DB_USER", raising=False)
    with pytest.raises(DatabaseConfigurationError):
        SessionFactory()
```

Fonte: Autoria própria, 2025

4.13 Vantagens de Implementação

I. Segurança
- Credenciais protegidas em variáveis de ambiente
- Validação prévia de configuração
- Transações ACID com rollback automático

II. Manutenibilidade
- Código limpo e organizado
- Separação de responsabilidades
- Padrões de projeto bem estabelecidos

III. Flexibilidade
- Suporte a múltiplos bancos de dados
- Configuração dinâmica
- Fácil adaptação para novos requisitos

IV. Robustez
- Tratamento adequado de erros
- Logging detalhado
- Health-check de conexão

4.14 Comparação com Abordagens Alternativas

Aspecto	SQLAlchemy ORM	SQL Raw	Outros ORMs
Produtividade	Alta	Baixa	Média
Segurança	Alta	Variável	Alta
Performance	Boa	Excelente	Boa
Manutenibilidade	Excelente	Baixa	Boa

4.15 Possíveis Melhorias

I. Em produção:
- Desabilitar echo=True
- Adicionar connection pooling
- Implementar retry mechanism

II. Monitoramento:
- Métricas de performance
- Logs estruturados
- Alertas de saúde

III. Segurança:
- Criptografia adicional
- Audit trails
- Backup automatizado

4.16 Conclusão

Esta implementação do SQLAlchemy no projeto MyApp demonstra uma arquitetura sólida e profissional para acesso a dados. A abordagem adotada proporciona:

- Segurança através de boas práticas de configuração
- Manutenibilidade via padrões de projeto estabelecidos
- Flexibilidade com suporte a múltiplos bancos
- Robustez com tratamento completo de erros

A solução se alinha perfeitamente com as melhores práticas de desenvolvimento Python e oferece uma base sólida para evoluções futuras do sistema, garantindo escalabilidade e confiabilidade no acesso aos dados.

5 ACESSIBILIDADE

A acessibilidade é um aspecto essencial no desenvolvimento de sistemas, garantindo que todos os usuários, incluindo aqueles com deficiências, possam interagir de maneira eficiente com o site. Com base na análise do painel de controle do sistema, listamos algumas funcionalidades que podem ser implementadas para melhorar a experiência do usuário, atendendo a diferentes necessidades de acessibilidade.

5.1 Objetivo da Implementação

O objetivo da implementação de recursos de acessibilidade no sistema é garantir que ele esteja disponível para um público amplo, incluindo usuários com deficiência auditiva, visual e motora. Com isso, o sistema deve permitir uma navegação mais intuitiva e personalizada, garantindo que usuários com diferentes tipos de necessidades possam utilizar as funcionalidades de forma confortável e eficaz.

5.2 Funcionalidades de Acessibilidade a Serem Implementadas

5.2.1 Ajuste de Tamanho de Fonte e Contraste:

- Para garantir que usuários com deficiência visual possam ler o conteúdo facilmente, deve ser oferecida a opção de aumentar o tamanho da fonte e ajustar o contraste das cores.
- Um botão de controle de fonte e contraste pode ser adicionado no painel lateral ou em um menu de acessibilidade no canto superior direito da página. O usuário pode escolher entre opções de "Fonte Maior" e "Modo de Alto Contraste", que alteram a legibilidade do conteúdo sem afetar a funcionalidade.

5.2.2 Tradução de conteúdo para libras

- Para atender usuários surdos ou com deficiência auditiva, o site deve fornecer uma tradução simultânea do conteúdo em Língua Brasileira de Sinais (Libras). Isso pode ser feito por meio de um avatar ou janela de tradução.
- Utilizar APIs como Vlibras, que oferecem tradução automática de textos para Libras. O avatar exibido no canto inferior ou em uma área destacada do painel pode sinalizar o conteúdo da página em tempo real.

5.2.3 Alertas e Notificações Acessíveis

- As notificações ou alertas exibidos no sistema, como o status de aprovação ou alteração de pedidos, devem ser visíveis para todos os usuários, incluindo aqueles com deficiência auditiva.
- Exibir as notificações de maneira clara e visível, como em banners ou caixas de diálogo, além de garantir que sejam acompanhadas de sons e/ou animações de destaque que possam ser percebidas por pessoas com deficiência auditiva.

5.2.4 Controle de Acessibilidade Personalizável

- Oferecer um controle de acessibilidade personalizável, no qual o usuário possa ativar ou desativar as opções de acessibilidade de acordo com suas necessidades.
- Incluir um menu flutuante no canto da tela que permita o controle das opções de "Fonte Maior", "Alto Contraste" e "Tradução para Libras". O usuário poderá ativar ou desativar essas funcionalidades com facilidade, de acordo com suas preferências.

5.3 Conclusão

A implementação dessas funcionalidades de acessibilidade vai garantir que o sistema seja mais inclusivo e acessível para uma gama maior de usuários. Com essas opções, atendemos a necessidades específicas, como a leitura de conteúdos, interação sem mouse e comunicação em Libras, tornando a plataforma mais amigável e funcional para todos.

6 CONSIDERAÇÕES FINAIS

6.1 Principais conclusões

O desenvolvimento do S.A.O.R.I. – Sistema Avançado de Organização de Recursos e Insumos representou uma oportunidade concreta de aplicar, em um contexto realista, os conhecimentos adquiridos ao longo do curso de Desenvolvimento de Software Multiplataforma. O sistema mostrou-se capaz de atacar o problema central identificado na pesquisa: a dificuldade de organizar listas de materiais, kits de componentes e ordens de compra de forma padronizada, rastreável e integrada aos fluxos de aprovação internos.

Com a combinação da aplicação desktop, da camada web e da API, o S.A.O.R.I. contribui para reduzir erros manuais, agilizar a consulta a fornecedores e apoiar o planejamento de materiais alinhado a conceitos de BOM e MRP, trazendo ganhos de eficiência e confiabilidade para o processo de gestão de recursos.

6.2 Recomendações

Para aprimorar ainda mais o sistema, recomenda-se:

- Aprimorar a experiência de uso no módulo web, com telas mais responsivas e componentes de interface que favoreçam o trabalho em diferentes resoluções e dispositivos.
- Ampliar os relatórios gerenciais, incluindo indicadores de consumo de materiais, tempo de aprovação, custos e histórico de uso de kits, apoiando decisões estratégicas.
- Investir na evolução contínua dos recursos de acessibilidade, com mais opções de contraste, personalização tipográfica e compatibilidade com leitores de tela.
- Estruturar uma suíte de testes automatizados mais ampla (unitários, integração e ponta a ponta) para facilitar a manutenção e reduzir riscos em futuras alterações.

6.3 Limitações do estudo

Entre as principais limitações do projeto, destacam-se:

- A validação do sistema em um ambiente predominantemente acadêmico, com cenários controlados e número reduzido de usuários, o que limita a observação de comportamentos em contextos industriais mais complexos.
- A ausência de um processo formal de implantação em empresas, incluindo migração de dados legados e integração com sistemas corporativos já existentes, como ERPs consolidados.
- O fato de a arquitetura atual ainda não explorar estratégias mais avançadas de escalabilidade e observabilidade, como monitoramento centralizado, métricas em produção e testes de carga em larga escala.

6.4 Perspectivas futuras

Para a continuidade e expansão do S.A.O.R.I., despontam as seguintes perspectivas:

- Integrar o sistema a ERPs e outros sistemas de chão de fábrica, permitindo troca automática de dados de ordens de produção, estoques e listas técnicas.
- Evoluir o módulo web para suportar múltiplas unidades ou plantas industriais, com controle de permissões mais refinado e workflows configuráveis por contexto.
- Incluir mecanismos de análise de dados e painéis em tempo real, explorando métricas de lead time, consumo de materiais e gargalos de aprovação.
- Considerar, em etapas futuras, o uso de conteinerização e outras estratégias de implantação quando o sistema atingir um estágio mais avançado de maturidade e demanda, mantendo no curto prazo o foco em estabilidade e validação funcional em campo.

REFERÊNCIAS

GRAZIANI, Álvaro Paz. ***Planejamento, programação e controle da produção: livro didático***. Florianópolis: Ânima Educação, 2021. Disponível em: https://repositorio.animaeducacao.com.br/items/1c428a0f-69c0-4428-b13f-6c9c8ccd0ae7. Acesso em: 12 set. 2025.

PITHON, A. J. C. **Projeto organizacional para a engenharia concorrente no processo de compras públicas**.2004. Disponível em: https://core.ac.uk/download/pdf/55602529.pdf. Acesso em: 13 set. 2025.

PRODANOV, C. C.; FREITAS, E. C. **Metodologia do trabalho científico:** Métodos e técnicas da pesquisa e do trabalho científico . 2. ed. Rio Grande do Sul: Universidade Feevale, 2013.

NETO, E. R. **Análise SWOT**: Planejamento Estratégico para Análise de Implantação e Formação de Equipe de Manutenção em uma Empresa de Segmento Industrial. 2011. 33p. Trabalho de Conclusão de Curso Graduação em MBA – Gestão estratégica da manutenção, produção e negócios – Faculdade Pitágoras. São joão del rei, 2011.

SENAI ES. ***Lean manufacturing*: o que é e como funciona?** Senai ES, 18 maio 2018. Disponível em: https://senaies.com.br/lean-manufacturing-o-que-e-e-como-funciona/. Acesso em: 12 set. 2025.

SEVERINO, A. J. **Metodologia do Trabalho Cientifico.** 22a ed. São Paulo: Cortez, 2002

SLACK, Nigel; CHAMBERS, Stuart; JOHNSTON, Robert**. Administração da produção**. 2. ed. São Paulo: Atlas, 2002.

TOLEDO, José Carlos de; LIMA, José Carlos de; OLIVEIRA, José Carlos de. A **model for the development of a manufacturing system**. *Gestão & Produção*, São Carlos, v. 3, n. 2, p. 3–16, 1996. DOI: 10.1590/S0104-530X1996000200003

FORMULÁRIO DE IDENTIFICAÇÃO

<table>
<tr><td colspan="4">DADOS DO RELATÓRIO TÉCNICO E/OU CIENTÍFICO</td></tr>
<tr><td colspan="3">Título e subtítulo: S.A.O.R.I – Sistema Avançado de Organização Recursos e Insumos</td><td>Tipo de relatório: Técnico-científico</td></tr>
<tr><td colspan="3">Instituição: Fatec Mauá</td><td>Data: 04/12/2025</td></tr>
<tr><td colspan="3">Curso: Desenvolvimento de Software Multiplataforma</td><td>Nota:</td></tr>
<tr><td colspan="4">Autor(es): Augusto de Barros Almeida, Bruno Henrique Gusmão Gonçalves, Renato Fernandes da Silva e Rodrigo Bastos Amaral</td></tr>
<tr><td colspan="4">Orientador: Luana Lourenço</td></tr>
<tr><td colspan="4">Banca avaliadora: Carlos Ronny de Souza
Ivan Carlos Pavão</td></tr>
<tr><td colspan="4">Produto desenvolvido: Sistema Avançado de Organização Recursos e Insumos.</td></tr>
<tr><td colspan="4">Aplicativo Mobile e Web desenvolvido utilizando Flutter. Ambiente de desenvolvimento: Android Studio. Interface construída parcialmente via FlutterFlow. Backend utilizando Supabase (autenticação, API REST, banco de dados Postgres).</td></tr>
<tr><td colspan="4">Palavras-chave/Descritores: Python, MRP, BOM, Lista de peças, Sistema de pedidos</td></tr>
<tr><td>Edição
1</td><td>Nº de páginas
34</td><td>Nº do volume/parte
1</td><td>Área do conhecimento:
Tecnologia da Informação</td></tr>
<tr><td colspan="4">Observações/notas</td></tr>
</table>

Fonte: Estrutura adaptada pela autora com base na norma ABNT NBR 10719:2015.