Aplicação de Sistema Preditivo em Gestão de Estoque direcionado à Oficinas Mecânicas de Pequeno Porte

Nelson de Almeida Machado Batista[1], Rafael Vieira Coelho[2]

[1]Instituto Federal do Rio Grande do Sul – Campus Farroupilha (IFRS)
Rodovia dos Romeiros, 567 – 95170000 – Farroupilha – RS – Brasil

@nelson.batista@aluno.ifrs.edu.br

Abstract. *Inventory management is a recurring challenge in auto repair shops, especially small ones. Generally, in these smaller establishments, inventory control is done manually or with poorly structured tools. This scenario results in a high probability of frequent problems, such as parts shortages, causing delays in service development and reduced customer satisfaction. Similarly, excess idle items generate costs and immobilize capital. In this context, this work proposes the development of an inventory forecasting system based on Artificial Intelligence applied to small auto repair shops. The solution integrates historical and operational data to estimate future parts consumption, identify shortage risks, and assist in decision-making. The results demonstrate that the use of predictive methods can offer greater accuracy in purchasing planning, reduce waste, and make the operation more competitive in the automotive market.*

Resumo. *A gestão de estoques é um desafio recorrente em oficinas mecânicas, especialmente nas de pequeno porte. Geralmente nesses estabelecimentos menores o controle é feito de forma manual ou com ferramentas pouco estruturadas. Esse cenário resulta em uma grande probabilidade de problemas frequentes, como ruptura de peças, causando lentidão no desenvolvimento dos serviços e redução da satisfação do cliente. Assim como excesso de itens parados, que gera custos e imobiliza capital. Diante desse contexto, este trabalho propõe o desenvolvimento de um sistema de previsão de estoques baseado em Inteligência Artificial aplicado a oficinas mecânicas de pequeno porte. A solução integra dados históricos e operacionais para estimar o consumo futuro de peças, identificar riscos de escassez e auxiliar na tomada de decisões. Os resultados demonstram que o uso de métodos preditivos pode oferecer maior precisão no planejamento de compras, reduzir desperdícios e tornar a operação mais competitiva no mercado automotivo.*

1. Introdução

As oficinas mecânicas têm como função desempenhar um papel fundamental no âmbito da manutenção e da conservação dos veículos. Desta forma, é garantida a segurança e o aumento da vida útil dos automóveis. Dentro desse cenário, se faz presente uma gestão eficiente dos recursos, no que se diz respeito ao estoque de materiais e peças. Trata-se de um fator determinante em respeito a competitividade no mercado e a qualidade do atendimento. Uma oficina mecânica que possui um controle adequado no estoque, possui facilidade para atender as demandas evitando atrasos, com agilidade e, desta maneira, conseguindo minimizar as perdas financeiras referentes a produtos indisponíveis ou obsoletos (Heizer & Render, 2017; Ballou, 2009).

De acordo com Ching (2010), a gestão de estoque compreende-se por controlar e planejar a entrada e saída dos materiais, realizando monitorias das épocas de reposição,

quantidades e pedidos. Para obter um ótimo gerenciamento, deve haver um equilíbrio entre oferta e demanda, foco em redução de custos em manutenção, obter espaço otimizado e certeza garantida que o produto fique disponível no momento adequado de seu uso. Porém, todo esse contexto se torna um desafio devido às demais variações no mercado que influenciam diretamente nos fatores econômicos, eventos inesperados e até mudanças sazonais. Toda essa instabilidade exige a implementação de estratégias flexíveis e um acompanhamento, com o intuito da certeza que os níveis de estoque e demanda estejam alinhados, dessa forma, evitando a ruptura em respeito ao excesso de produtos.

A inteligência artificial (IA), se apresenta no mundo moderno como uma tecnologia que possui grandes capacidades de transformar radicalmente os processos logísticos e operacionais. Ela se define como uma ferramenta na ciência da computação que possui dedicação especial nos sistemas que realizam tarefas que requerem inteligência humana (Russel & Norvig, 2020). Ela possui abrangência em técnicas como redes neurais, aprendizados de máquinas e análises preditivas que possuem análise de um grande volume de dados, prever eventos futuros e identificar padrões (Taniwaki et al., 2024).

A utilização da inteligência artificial em um ambiente empresarial, segundo Pereira Júnior (2003), eleva a eficiência em âmbitos operacionais e compreensivos perante padrões de consumo. Dessa forma, permite-se elevar as estratégias e torná-las o mais assertivo possível. Com aplicação à gestão de estoques, a inteligência artificial tem potencial de ser bem treinada a prever com precisão demandas de produtos, reduzir ao máximo erros humanos, deixar otimizado os níveis de reposição e tornar os processos logísticos mais coordenados e inteligentes possível (Kimes,2018; Min,2010). Tais ferramentas proporcionam que as empresas realizem os ajustes de estoque e compras com mais segurança e principalmente reduzir custos desnecessários.

Nas empresas de pequeno porte, referindo-se estritamente às oficinas mecânicas, observa-se a presença de um desafio acentuado de otimizar os processos e manter a sustentabilidade em larga escala. Ademais, a transformação digital nesse setor garante ganhos na concorrência, eficiência e qualidade (Gerhardt, s.d; Zimmerer apud Beraldi & Filho, 2000).

Frente a essa realidade, este trabalho propõe o desenvolvimento e a validação de um sistema web preditivo para a gestão de estoques em oficinas mecânicas, utilizando técnicas de Inteligência Artificial e visualização de dados. A solução tem como propósito demonstrar os benefícios e desafios da aplicação de métodos preditivos no cotidiano operacional de empresas de pequeno porte, oferecendo maior precisão, confiabilidade e agilidade nas decisões de reposição de peças. Dessa forma, busca-se fornecer aos gestores uma ferramenta inteligente capaz de minimizar riscos de escassez ou excesso de itens, otimizar o planejamento de compras e elevar a eficiência das atividades no setor automotivo. Para alcançar esse propósito, foram definidos os seguintes objetivos específicos:

Estruturar uma base de dados mensal a partir de registros históricos de consumo e estoque de peças automotivas;

Desenvolver uma aplicação web interativa utilizando Python e Flask, integrando bibliotecas como Pandas, Plotly e Scikit-learn para análise e visualização de dados;

Implementar um modelo de regressão linear para gerar previsões de consumo em um horizonte de até doze meses;

Permitir a visualização comparativa entre consumo real e previsto, destacando cenários de escassez e excesso de materiais;

Aplicar indicadores e classificações automáticas de status do estoque (crítico, normal, ruptura e excesso);

Desenvolver um módulo de recomendações inteligentes, com base nas previsões e métricas calculadas, para apoiar decisões de compra e reposição;

A partir desses objetivos, este trabalho busca demonstrar de forma prática como a aplicação de técnicas de inteligência artificial pode contribuir para a gestão de estoques em oficinas mecânicas. O desenvolvimento do sistema proposto visa unir a simplicidade operacional necessária a pequenas empresas com o potencial analítico das tecnologias preditivas, oferecendo uma ferramenta capaz de transformar dados históricos em informações úteis para a tomada de decisão.

2. Referencial Teórico

A inteligência Artificial (IA) é definida como uma aplicabilidade da execução de tarefas em sistemas computacionais, que de certa forma, historicamente demanda da inteligência humana - aprender com dados, aceitar padrões e decidir sobre incertezas (Taniwaki et al., 2024). Em alinhamento dessa visão, McCarthy (2007) enfatiza a inteligência artificial como uma ciência e a engenharia capazes de construir maquinário inteligente, com ênfase em aplicações que encenam os processos cognitivos humanos auxiliando as tomadas de decisões.

Essa determinada área da inteligência artificial possui alguns marcos históricos. A famosa pergunta “As máquinas podem pensar?” Construída por Alan Turing em meados dos anos 1950, seguido do Teste de Turing, juntos deram permissão para caminhos novos à agenda científica para avaliar o comportamento inteligente. Em 1956, foi estabelecido o termo “Inteligência Artificial” , classificado como disciplina formal. Esse intusiasmo inicial deu segmento aos períodos da “inverno de IA” - estabeleceu-se limitações tecnológicas e expectativas não correspondidas, dessa forma interrompendo avanços -, porém todo o progresso foi reativado e assim aplicando as aplicações práticas enfatizando todo o poder computacional e evolução de técnicas como as redes neurais e aprendizado das máquinas (Taniwaki et al., 2024; Russel & Norvig, 2020).

A IA tem apresentado muita eficiência em termos operacionais e de inteligência de mercado. Pereira Junior (2023) avalia que a tecnologia possui um melhor desempenho em análises mais detalhadas de consumo e comportamento e dessa forma sendo mais assertiva. Adicionalmente, Tenés Trillo (2023) ressalta três frentes de impacto:

A análise de dados possui uma precisão maior referente às decisões estratégicas;

Automação de tarefas repetidas, dessa forma liberando o usuário para funções de maior valor;

Diferenciação de experiências com um apontamento direto na satisfação e conquista de clientes.

Essa determinada forma de gestão possui um lugar especial na cadeia dos suprimentos, justamente por ela lidar com administração e controle de recursos, com foco na busca da sincronização entre custo e disponibilidade (Heizer & Render, 2017). Na prática, administrar estoques envolve determinar 3 fases - O que - Quanto - e - Quando - repor do modo que o produto esteja concedido no momento adequado, com foco em reduzir os riscos de ruptura e garantir capital imobilizado (Heizer & Render, 2017).

Ching (2010) classifica a definição de gestão de estoques como um planejamento e monitoramento de entrada e saída dos materiais com foco em épocas de reposição, quantidades e pedidos. Toda essa visão operacional é atrelada à visão de custo total sugerido por Ballou (2009), que o mesmo recomenda levar em consideração além de todo custo de manutenção, custos de pedidos e risco de ruptura, dessa forma, ajustando todos os níveis de estoque de acordo com previsões de demanda e as variações de mercado e somado a isso suporte de técnicas definidas como os pontos de reposição e categorias de serviço.

A gestão de estoques sempre desempenhou papel estratégico nas organizações, atuando de forma direta na eficiência operacional e na redução de custos. Conforme Chaves Júnior (2023), esse controle é especialmente relevante em setores como o automotivo, nos quais a disponibilidade imediata de peças impacta diretamente o desempenho produtivo e o tempo de atendimento ao cliente. Gomes (2024) complementa que, historicamente o gerenciamento de estoques se apoiou em modelos quantitativos clássicos, criados com o propósito de equilibrar custos de aquisição, armazenagem e reposição, estabelecendo a base teórica para as práticas modernas de planejamento de materiais.

Entre os principais modelos, destacam-se o EOQ (Economic Order Quantity), o Ponto de Pedido e a Curva ABC, aplicados em diferentes tipos de negócio. Chaves Júnior (2023) explica que esses métodos são eficazes para priorizar itens conforme a relevância econômica e frequência de uso, permitindo decisões de compra mais assertivas e redução de custos operacionais. A Curva ABC, em especial, possibilita classificar os itens de estoque segundo sua representatividade no custo total, garantindo maior controle sobre os produtos de maior impacto financeiro.

Segundo Gomes (2024), os modelos quantitativos clássicos como o EOQ proposto por Ford W. Harris (1913), formaram o alicerce para o surgimento dos primeiros sistemas de previsão de demanda. Estes modelos buscavam determinar a quantidade ideal de pedido e o momento exato para reposição, minimizando custos de armazenagem e evitando a falta de produtos. No entanto, com o aumento da complexidade das cadeias de suprimentos e a necessidade de respostas ágeis, surgiu a demanda por soluções mais dinâmicas e adaptáveis, o que abriu espaço para a Inteligência Artificial (IA) como ferramenta de aprimoramento desses métodos tradicionais.

O desenvolvimento do projeto da previsão de estoques foi construído através de um conjunto de tecnologias voltadas para o desenvolvimento web e ciência de dados. A escolha das ferramentas priorizou a integração entre simplicidade, escalabilidade e

compatibilidade, consequentemente, garantir que a aplicação fosse acessível e tecnicamente robusta.

3. Metodologia

O desenvolvimento desse sistema tem como objetivo principal realizar a previsão mensal através do consumo das peças em uma oficina mecânica, com a presença da base de dados históricos registrados em arquivos CSV. Além disso, o sistema integra esses dados com informações operacionais mantidas em JSON.

O projeto foi construído com a utilização da linguagem Python e o microframework Flask, dessa forma, surgiu a criação da interface web cujo manuseio é simples e funcional e acessível para qualquer navegador.

O gestor da oficina pode acompanhar os relatórios interativos, consumos das peças e visualizar previsões. Todas informações são passadas para um processamento por um modelo de aprendizado de máquina baseado em regressão linear que tem como função estimar o futuro contendo uma base em padrões históricos.

Para ter uma visão geral do sistema, ele foi dividido em quatro camadas, sendo elas, interface e visualização, processamento e regras de negócio, modelagem preditiva e persistência de dados. Confira a Figura 1.

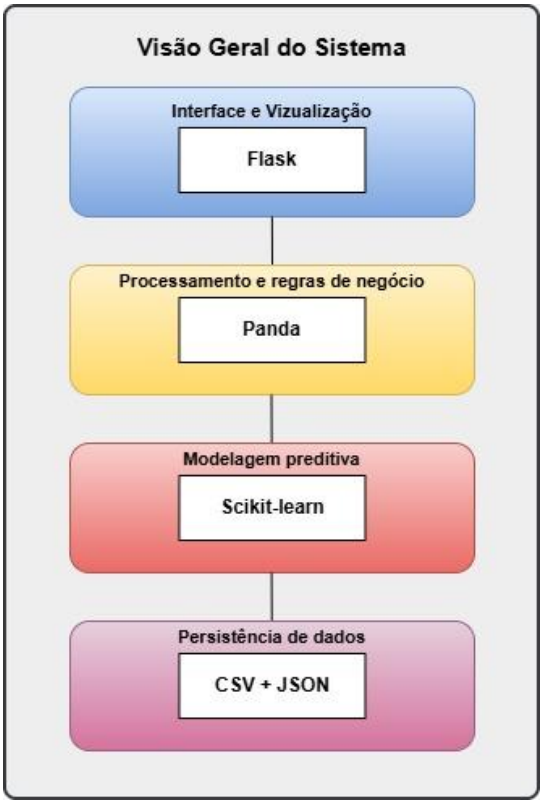

Figura 1: Visão Geral do Sistema

A primeira camada refere-se à interface e visualização, direcionada como o ponto principal de interação entre usuário e sistema. Nessa parte que ocorre a comunicação entre o gestor da oficina e o sistema, dessa forma permitindo o envio das requisições, acesso a navegação das páginas e visualização das informações geradas. No desenvolvimento, essa etapa foi implementada com o Flask, que possui a função de organizar e gerenciar as rotas de acesso, tratar as requisições HTTP e gerar o conteúdo dinâmico. O Flask também tem a responsabilidade de atuar em conjunto dos mecanismos de templates Jinja2, dessa forma integrando os dados gráficos criados no back-end com função direta nas páginas HTML. Assim, o projeto oferece uma ótima experiência interativa clara e com um visual intuitivo.

Na segunda camada, processamento e regras de negócio, se responsabiliza pelo conjunto de tratamento, manipulação e organização dos dados no sistema. Para isso é utilizado a biblioteca Pandas, que é responsável por carregar as informações dos registros históricos em CSV, convertendo as estruturas do tipo DataFrame. Nessa etapa são executadas operações como conversões de tipos, ordenações e agregações estatísticas. Em paralelo, é realizada nesta camada a utilização de informações armazenadas em JSON, que representam o estoque atual manipulado pelo usuário, Assim, o sistema consegue combinar de forma consistente, dados históricos (CSV) e dados operacionais que são atualizados em tempo real (JSON). Além disso, esta camada implementa as regras de negócio, como classificação automática de status das peças e cálculo dos indicadores utilizados na etapa de previsão.

A terceira camada se apresenta como a modelagem preditiva. Nela é implementado o modelo de Regressão Linear, através da biblioteca Scikit-learn, finalidade presente definida como o consumo futuro de peças, baseado nos dados históricos registrados. Tal modelo de aprendizagem tem como objetivo buscar padrões de comportamento no histórico de consumo mensal, traçando um mapeamento entre a relação conjunta entre tempo e demanda. Nessa relação o sistema consegue gerar previsões para meses futuros, assim permitindo ao usuário antecipar as necessidades de reposição e consequentemente planejar num cenário aliviado e positivo às compras.

Na camada final temos a persistência de dados, que é responsável pelo armazenamento e manutenção das informações processadas e históricas. Aqui como principal meio de persistência são utilizados os arquivos CSV, escolha que prioriza a compatibilidade e simplicidade, assim dispensando a necessidade de um banco de dados relacional. Abordagem mais destinada à oficinas de pequeno porte, que precisam de uma solução mais leve e com facilidade na utilização. Nessa estrutura o arquivo CSV definido como a base de dados para leitura e análise, enquanto isso novos dados podem ser adicionados de forma simples, tornando o sistema escalável e acessível.

3.1. Modelagem e Requisitos

Para facilitar a compreensão do funcionamento da aplicação desenvolvida, este tópico apresenta de forma estruturada os principais componentes do sistema. Foi projetada de primeiro momento a modelagem de casos de uso, representando de uma visão geral as principais interações entre o sistema e o usuário. Neste projeto é presente o ator principal que podemos classificar como Gestor da Oficina, este que é responsável pelo

acesso da plataforma, visualizar os dados de estoque e por fim gerar as previsões de consumo das peças registradas.

A modelagem foi construída baseada nas ações reais que um determinado gestor de oficina executa dentro da aplicação, desde o momento em que o sistema é aberto até na concessão das previsões e recomendações de estoque. O diagrama é apresentado na Figura 2, ilustrando graficamente de forma simples e objetiva as interações, com o usuário e funcionalidades do sistema. Ele representa as principais interações do gestor da oficina com o sistema de previsão de estoque. Inicialmente, o gestor acessa o sistema, etapa responsável por permitir a entrada na aplicação e habilitar o uso das demais funcionalidades. Uma vez dentro da plataforma, o usuário tem a possibilidade de visualizar o dashboard, no qual são exibidos os indicadores gerais do estoque e informações sintetizadas sobre o consumo e o status das peças.

Outra funcionalidade importante consiste na opção de selecionar uma peça específica, permitindo ao sistema carregar e organizar os dados históricos e atuais referentes ao item escolhido. A partir dessa seleção, o usuário pode gerar a previsão de consumo, processo em que o modelo preditivo aplica técnicas de regressão para estimar a demanda futura. Após a geração da previsão, o sistema disponibiliza a visualização do gráfico correspondente, exibindo de forma comparativa o consumo real e as estimativas calculadas, facilitando a interpretação.

O sistema também conta com um módulo de gerenciamento de estoque, no qual são apresentados indicadores operacionais baseados nas regras de negócio, como níveis críticos, ruptura, excesso e normalidade. Por fim, o usuário pode acessar o histórico das peças, obtendo uma visão detalhada da evolução do consumo ao longo do tempo, o que contribui para análises mais completas e tomadas de decisão fundamentadas.

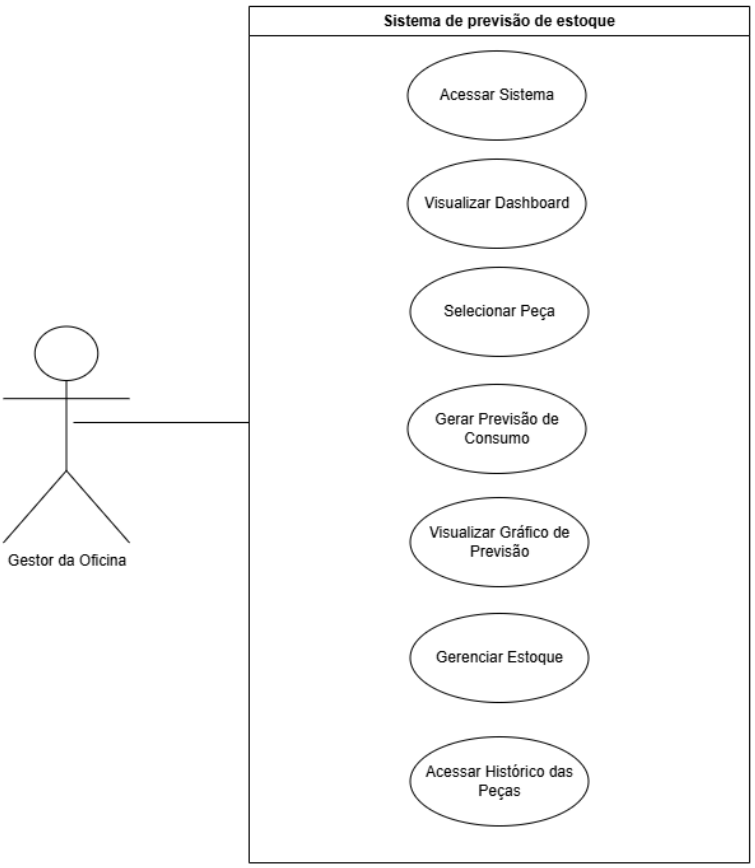

Figura 2. Diagrama de Caso de Uso

Com base no diagrama de caso de uso apresentado anteriormente, tornou-se possível identificar e descrever as principais funcionalidades que o sistema de previsão de estoque deve oferecer ao gestor da oficina. Nesta etapa, é traçado o objetivo de detalhar de uma forma organizada e clara os comportamentos que o sistema deve oferecer.

Nas tabelas a seguir (tabela 1 e tabela 2) serão apresentados os requisitos funcionais e não funcionais suas respectivas descrições de cadas funcionalidade.

Tabela 1. Requisitos funcionais

Acessar o sistema via navegador	Permitir que o gestor possa acessar o Sistema em qualquer navegador
Visualizar o dashboard principal	Exibir um painel interativo com gráficos De consumo e estoque das peças cadastradas
Selecionar peça para análise	Usuário possa escolher uma peça para Visualizar dados de consumo mensal
Gerar previsões de consumo	Utilizar o método de aprendizado de Máquina (Regressão Linear) para gerar o consumo futuro
Comparativo entre consumo real e previsto	Ilustração de forma gráfica o histórico e as previsões de consumo, destacando tendências
Apresentar recomendações de estoque	Sugerir ações de compra ou reposição baseadas em resultados do modelo e nas métricas de risco
Atualizar dados a partir de arquivos CSV e JSON	Permitir a leitura de um arquivo CSV (dados históricos) e JSON (estoque atual) contendo a sincronia de informações de estoque e consumo
Classificar status do estoque	Classificar cada item como “crítico”, “ruptura”, “normal” ou “excesso” de acordo com as métricas calculadas

Tabela 2. Requisitos não funcionais

Usabilidade	Interface simples e intuitiva, acessível a usuários que não possuam conhecimento técnico avançado
Portabilidade	Acesso via navegador em diferentes sistemas operacionais
Desempenho	Carregamento dos dados e geração dos gráficos devem ocorrer de forma rápida e estável
Confiabilidade	O sistema deve processar dados históricos consistentes, sem perdas ou duplicações
Manutenibilidade	Código modular, organizado por camadas (rotas, modelos ou duplicações)
Escalabilidade	A estrutura deve permitir futura integração com banco de dados e novos modelos preditivos
Segurança de dados	Os dados do arquivo CSV e JSON devem ser processados localmente, garantindo confidencialidade dos dados
Compatibilidade tecnológica	O sistema deve operar corretamente com Python e bibliotecas Flask, Pandas e Plotly

‘

Dessa forma, os requisitos definidos serviram como uma base importante para o planejamento do desenvolvimento da aplicação. Elementos que foram guia para a implementação das funcionalidades e definição de estratégias técnicas aplicadas ao longo do processo na construção do sistema.

3.2. Organização dos Dados

Antes da implementação do sistema, foi necessário construir a base de dados histórica que servirá como insumo para os cálculos de previsão. Para isso, foram utilizados dois conjuntos de dados públicos disponíveis na plataforma Kaggle: o “Logistics and Supply Chain Dataset” (DATASETENGINEER, 2023) e o “Vehicle Maintenance Record” (NAVINS7, 2023).

O primeiro Dataset contém registros de movimentações logísticas, níveis de estoque, fornecedores e pedidos, enquanto o segundo reúne informações sobre manutenções automotivas e consumo de peças em reparos. Ambos foram integrados e ajustados para gerar o arquivo em CSV, definido como o dataset deste projeto. Cada linha representa um registro mensal de consumo e estoque de uma peça específica, permitindo ao modelo de Regressão Linear identificar padrões temporais e projetar o comportamento futuro do estoque. A base contempla doze meses de histórico consolidado, expandindo diferentes categorias de peças automotivas.

O processo de manipulação e limpeza dos dados foi realizado através do Python 3.11, utilizando bibliotecas como Pandas e NumPy para leitura, transformação e estruturação dos registros e para finalizar o Plotly para a geração dos gráficos interativos. A aplicação foi desenvolvida através do framework Flask que integra os componentes de análise e visualização na plataforma web.

Sobre a arquitetura de dados presentes nesse sistema segue o princípio da persistência híbrida, classificada com duas camadas. A estrutura foi planejada para garantir simplicidade, eficiência e atualização constante das informações.

A primeira camada, armazenada em arquivo CSV, é responsável por preservar os registros de consumo e estoque de peças, servindo de base para os cálculos de previsão. Na segunda camada, mantida em arquivo JSON, reflete o estado atual do estoque em tempo real, permitindo que as informações exibidas na interface web estejam sempre alinhadas à realidade da operação.

A separação entre a primeira camada e a segunda, permite equilibrar estabilidade e dinamismo: o CSV garante consistência e histórico de aprendizado, enquanto o JSON assegura atualizações imediatas e flexibilidade para as movimentações de estoque. Ademais, essa estrutura evita que alterações operacionais interfiram nos dados históricos, preservando a integridade das informações usadas nos cálculos de previsão.

O fluxo de informação entre as duas camadas (histórica) CSV e (operacional) JSON seguem uma sequência lógica e contínua para que o sistema funcione de forma integrada. De início os dados da camada histórica passam por um processo de tratamento e organização. O realizado uma conversão dos formatos de data (coluna mouth para o tipo datetime), realiza ordenações cronológicas e consolida os registros mensais de consumo e estoque de cada peça. Na sequência, esses dados são enviados para a camada de previsão, ali o modelo de aprendizado baseado em regressão linear, analisa os padrões históricos e gera estimativas de consumo futuro.

Os resultados obtidos da primeira camada são integrados à camada operacional, que utiliza o JSON para refletir o estoque atual. Através dessa combinação é realizado a comparação da previsão gerada pelo modelo com a situação real do estoque, utilizando métricas como nível de cobertura, ponto de reposição e status da peça (normal, risco, crítico ou excesso).

4. Desenvolvimento do Sistema

Uma funcionalidade muito importante do projeto é o dashboard de estoque, essa parte é importante, pois ali é exibido toda parte analítica de gráficos, tabelas sobre alguns dados das peças e status por quantidade de peças. Na primeira parte do Dashboard temos o resumo executivo, ali são apresentados os indicadores do sistema, confira a Figura 3 abaixo:

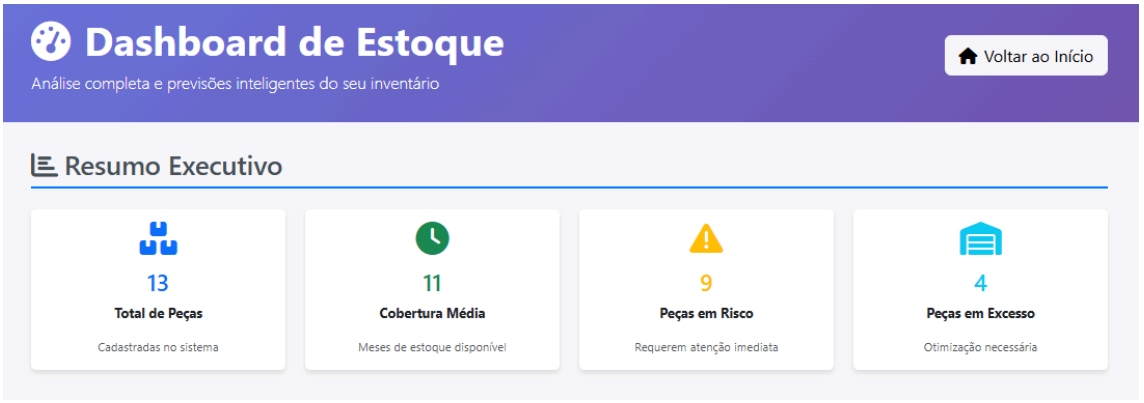

Figura 3: Resumo executivo de estoque

Se faz presente nesta primeira parte quatro tipos de indicadores, são eles:

- Total de Peças: Número total de itens registrados no estoque;
- Cobertura Média: Tempo em meses que o estoque duraria com base no consumo total;
- Peças em Risco: Itens classificados como críticos ou em ruptura;
- Peças em Excesso: Itens com um volume acima da demanda projetada.

Todos os indicadores são calculados de forma dinâmica pelo back-end, com a utilização do Pandas, dessa forma garantindo que os valores exibidos estejam sempre sincronizados com as últimas movimentações registradas no sistema

4.1 Análise por Peça

A funcionalidade de análise por peça presente no sistema tem como principal objetivo fornecer uma visão detalhada do comportamento individual de cada item do estoque, permitindo que o gestor compreenda não apenas o nível atual de disponibilidade, mas também a tendência de consumo e a previsão baseada nos dados registrados. A presente análise é fundamental com o intuito de apoiar as decisões de compra, reposição e otimização de armazenamento.

Nesta parte da seção abordarei a situação de status “Normal”, quando o item está dentro da faixa considerada ideal, sem risco de ruptura.

O sistema oferece um campo com todas as peças, assim oferecendo análise de todas as opções de peças, confira na imagem abaixo:

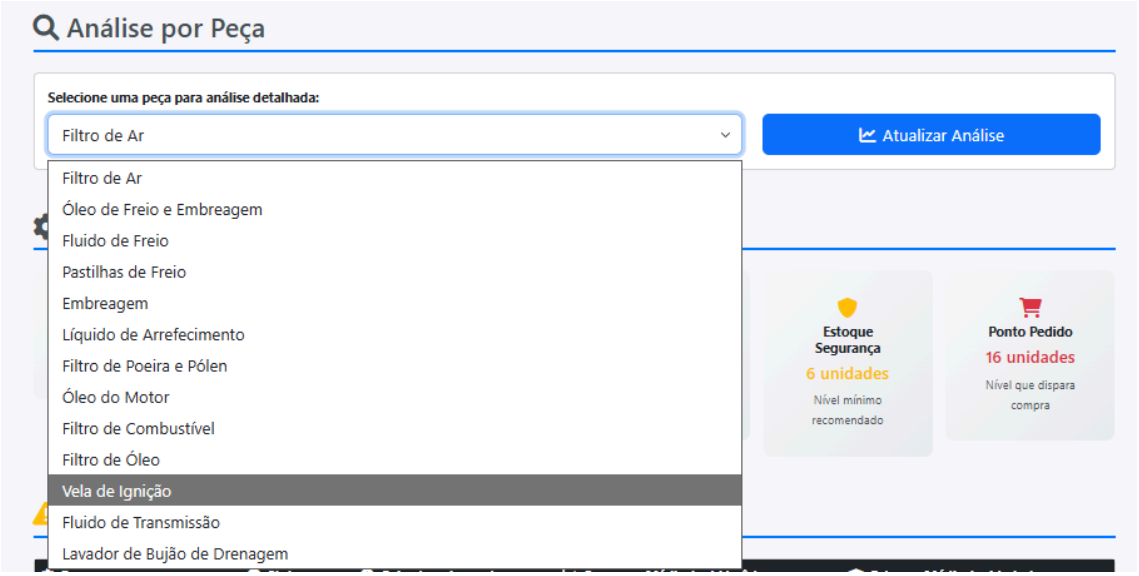

Figura 4: Seleção de peça para análise

4.1.1 Gráfico de Previsão

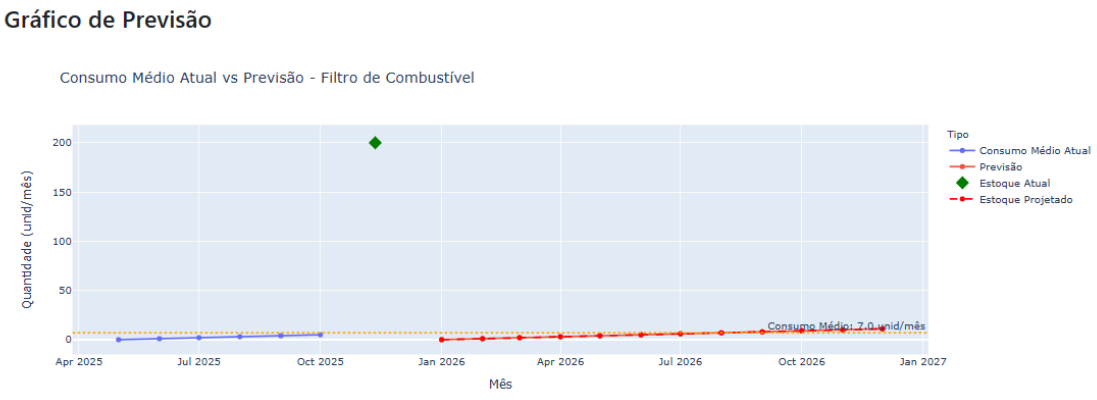

Figura 5: Estoque saudável

Ao visualizar a Figura 5, nos deparamos com um gráfico de previsão - “Consumo Médio Atual vs Previsão”. Ele aparece no sistema com o objetivo principal de comparar o consumo médio atual com a previsão futura de consumo e a evolução do estoque da peça selecionada.

No exemplo presente na Figura 5, nos deparamos que é exibido o item “Filtro de Combustível”, demonstrando o comportamento do consumo histórico e projetado,

indicando de forma visual como o estoque tende a se comportar ao longo dos meses seguintes.

O gráfico apresenta quatro legendas:

- Consumo médio atual: Representa a média histórica de consumo mensal da peça, servindo como referência de demanda real observada;
- Previsão: Indica o consumo futuro previsto pelo modelo de previsão implementado, utilizando os dados históricos da pela para estimar tendências;
- Estoque atual: Marca o ponto em que se encontra o nível de estoque disponível no momento da análise;
- Estoque Projetado: Mostra a evolução estimada do estoque ao longo do tempo, considerando o consumo previsto e a ausência ou presença de reposições.

Na ilustração do gráfico da Figura 5, observa-se o comportamento de consumo da peça "Filtro de Combustível" ao longo do tempo, combinando dados históricos e projeções geradas pelo modelo preditivo baseado em inteligência artificial implementado no sistema.

A Inteligência Artificial atua na etapa de previsão de consumo, processando os registros históricos da peça e aplicando técnicas de aprendizado estatístico para estimar o comportamento futuro da demanda. Esse processo origina a curva laranja, denominada "Previsão", que representa a estimativa de consumo para os próximos meses.

Constata-se que o consumo médio atual se mantém em torno de sete unidades por mês, caracterizando uma demanda estável. A curva de previsão segue essa mesma tendência, indicando que o modelo de IA reconheceu um padrão de estabilidade no histórico e projetou a continuidade desse comportamento.

O estoque atual, representado pela marca verde, encontra-se em um nível muito superior à demanda mensal, garantindo ampla disponibilidade do item. Já a linha de estoque projetado (traço vermelho) apresenta uma leve redução gradual ao longo do tempo, coerente com o consumo previsto e sem atingir níveis de risco.

Esse comportamento evidencia um cenário de equilíbrio e segurança, no qual o estoque está dimensionado de forma eficiente em relação à demanda prevista. Assim, o modelo de inteligência artificial contribui diretamente para a tomada de decisão gerencial, permitindo ao sistema antecipar tendências, evitar rupturas e otimizar o capital imobilizado em estoque. Esse exemplo representa uma estabilidade de consumo e estoque saudável dessa forma projetando um cenário de segurança operacional.

4.2.1 Gráfico de Previsão - Estoque em Ruptura

Como ilustração dessa seção observe a Figura 6, o gráfico demonstra de forma clara o nível de estoque tende a se esgotar ao longo dos meses seguintes, caso não sejam realizadas ações de reposição.

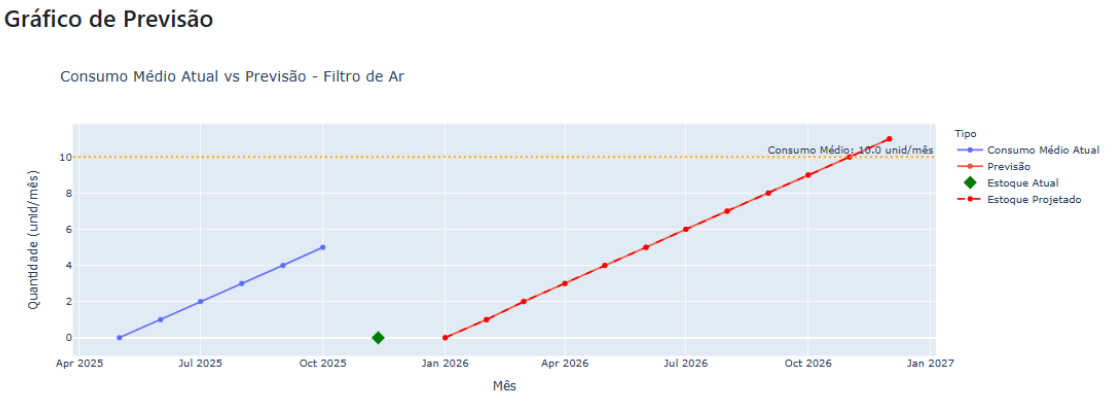

Figura 6: Estoque em Ruptura

A curva azul representa o consumo médio atual, calculado com base no histórico de utilização da peça. Nota-se que há uma tendência de crescimento gradual do consumo, o que indica um aumento na demanda ao longo do tempo. A linha marcada em vermelho representa o estoque projetado, ela tem começo em um ponto baixo e decresce rapidamente, atingindo o nível zero nos primeiros meses do período previsto. Essa interseção entre a linha de estoque projetado e o eixo horizontal evidencia o ponto de ruptura, ou seja, o momento em que o sistema prevê a ausência total da peça no estoque.

O marcador em verde, corresponde ao estoque atual, indica que a quantidade disponível no início da previsão já se encontra próxima de zero. Esse fator reforça a necessidade de reposição imediata, pois o sistema calcula que, mantido o ritmo de consumo total, o estoque não será suficiente para atender a demanda.

Na linha em laranja, temos representado o consumo médio de referência, como um indicador visual de comparação entre o consumo real e o previsto. No gráfico percebe-se que o consumo projetado ultrapassa a média de referência, assim contribuindo para o rápido esgotamento do estoque. Esse comportamento é resultado do processamento realizado pela função de previsão do sistema, que aplica o modelo de regressão linear, para estimar o consumo futuro e recalcular o estoque projetado de forma mensal.

A cada iteração, o modelo subtrai o consumo previsto de quantidade disponível e, ao identificar um valor menor ou igual a zero, o sistema registra automaticamente um alerta de ruptura. Olhando esse cenário de um ponto de vista analítico, ele mostra a efetividade da IA em detectar situações críticas de forma antecipada, possibilitando que o gestor visualize de forma clara a tendência de escassez e adote medidas corretivas, como a compra ou reposição de peças antes que o estoque acabe.

O gráfico apresentado na Figura 6 exemplifica a capacidade preditiva do sistema em identificar comportamentos de risco e apoiar a tomada de decisão baseada em dados.

4.2.2 Integração para Geração do Gráfico

O processo de geração do gráfico de previsão é implementado na função predict(). Este tem como finalidade integrar o histórico de consumo da peça, as previsões calculadas

pela IA e a geração do gráfico “Consumo Médio Atual vs Previsão”. A seguir, os trechos de código apresentam as principais etapas dessa função, desde a preparação dos dados até a renderização final do gráfico no front-end.

Na figura 7, contida abaixo, ilustra o início da função, onde são lidos os dados enviados pelo formulário (selected_piece), carregando o histórico da peça e iniciando o serviço de estoque. Nessa parte o sistema também garante que o Dataset histórico e o estoque atual sejam atualizados antes de prosseguir com o cálculo preditivo.

```python
@app.route("/predict", methods=["POST"])
def predict():
    selected_piece = request.form.get("piece")

    if not selected_piece:
        return "Nenhuma peça selecionada.", 400

    previsao_df = prever_consumo(df, selected_piece, meses=12)

    if previsao_df is None:
        return f"Nenhum dado disponível para a peça {selected_piece}.", 404

    # Histórico real da peça
    historico = df[df['part'] == selected_piece].sort_values('month')

    # Obter estoque atual real do sistema (não do CSV histórico)
    from .services.estoque_service import EstoqueService
    import os
    BASE_DIR = os.path.abspath(os.path.join(os.path.dirname(__file__), '..'))
    data_file = os.path.join(BASE_DIR, 'data', 'dataset_previsao_estoque_mensal_2025.csv')
    estoque_file = os.path.join(BASE_DIR, 'data', 'estoque_atual.json')
    estoque_service = EstoqueService(data_file, estoque_file)

    # Sempre recarregar estoque para garantir dados atualizados
    estoque_service.recarregar_estoque()
```

Figura 7. Leitura e preparação dos dados

Na Figura 8, contida abaixo, ilustra o início da função, onde são lidos os dados enviados pelo formulário (*selected_piece*), carregando o histórico da peça e iniciando o serviço de estoque. Nessa parte o sistema também garante que o *Dataset* histórico e o estoque atual sejam atualizados antes de prosseguir com o cálculo preditivo.

```python
@app.route("/predict", methods=["POST"])
def predict():
    selected_piece = request.form.get("piece")

    if not selected_piece:
        return "Nenhuma peça selecionada.", 400

    previsao_df = prever_consumo(df, selected_piece, meses=12)

    if previsao_df is None:
        return f"Nenhum dado disponível para a peça {selected_piece}.", 404

    # Histórico real da peça
    historico = df[df['part'] == selected_piece].sort_values('month')

    # Obter estoque atual real do sistema (não do CSV histórico)
    from .services.estoque_service import EstoqueService
    import os
    BASE_DIR = os.path.abspath(os.path.join(os.path.dirname(__file__), '..'))
    data_file = os.path.join(BASE_DIR, 'data', 'dataset_previsao_estoque_mensal_2025.csv')
    estoque_file = os.path.join(BASE_DIR, 'data', 'estoque_atual.json')
    estoque_service = EstoqueService(data_file, estoque_file)

    # Sempre recarregar estoque para garantir dados atualizados
    estoque_service.recarregar_estoque()
```

Figura 8. Leitura e preparação dos dados

Na próxima etapa (Figura 9) o código calcula as métricas da peça selecionada, como consumo médio, ponto de pedido e estoque de segurança. E simula o comportamento do estoque nos meses seguintes. Essa etapa é importante para gerar o valores que serão exibidos no gráfico de estoque projetado.

```
    # Calcular métricas da peça selecionada usando estoque atual
    metricas_peca = calcular_metricas_estoque_atual(df, selected_piece, estoque_service)
    if not metricas_peca:
        # Fallback para método antigo se não conseguir calcular
        metricas_peca = calcular_metricas_estoque(df, selected_piece)
        if metricas_peca:
            # Obter estoque atual real da peça selecionada
            estoque_info_selecionada = estoque_service.obter_estoque_atual(selected_piece, recarregar=True)
            estoque_atual_selecionada = estoque_info_selecionada.get('quantidade', 0) if estoque_info_selecionada else 0

            # Atualizar com estoque real
            metricas_peca['estoque_atual'] = estoque_atual_selecionada
            metricas_peca['cobertura_meses'] = estoque_atual_selecionada / metricas_peca['consumo_medio']
            if metricas_peca['consumo_medio'] > 0 else 0

            # Recalcular status baseado no estoque atual
            status, cor_status = calcular_status_peca(estoque_atual_selecionada, metricas_peca['consumo_medio'])
            metricas_peca['status'] = status
            metricas_peca['cor_status'] = cor_status

    # Calcular KPIs gerais usando estoque atual
    kpis_gerais = calcular_kpis_gerais_atualizados(df, estoque_service)
    return render_template("dashboard.html",
                           grafico1=grafico1_json,
                           grafico2=grafico2_json,
                           all_parts=all_parts,
                           selected_piece=selected_piece,
                           selected_piece_traduzido=selected_piece_traduzido,
                           metricas_peca=metricas_peca,
                           kpis_gerais=kpis_gerais)

from app.models.previsao import prever_consumo
```

Figure 9. Cálculos de métricas e simulação do estoque

Concedendo sequência às etapas, a seguinte verifica condições críticas, como a possibilidade de ruptura ou nível de estoque abaixo do estoque de segurança. Na Figura 10 são criados os dados históricos dos últimos meses para comparar com as previsões geradas pelo modelo de IA.

```python
# Detectar alertas de ruptura
alertas = []
for i, row in previsao_df.iterrows():
    if row['estoque_projetado'] <= 0:
        alertas.append({
            'mes': row['month'],
            'tipo': 'Ruptura',
            'severidade': 'Crítico',
            'mensagem': f"Estoque pode zerar em {row['month'].strftime('%Y-%m')}"
        })
    elif row['estoque_projetado'] < metricas_peca['estoque_seguranca']:
        alertas.append({
            'mes': row['month'],
            'tipo': 'Risco',
            'severidade': 'Alto',
            'mensagem': f"Estoque abaixo do nível de segurança em {row['month'].strftime('%Y-%m')}"
        })

# Criar dados baseados no estoque atual e consumo médio real
consumo_medio_real = metricas_peca['consumo_medio'] if metricas_peca else 0

# Criar dados para os últimos 6 meses (baseado no estoque atual)
from datetime import datetime, timedelta
import pandas as pd

# Gerar últimos 6 meses
data_atual = datetime.now()
meses_passados = []
for i in range(6, 0, -1):
    data_mes = data_atual - timedelta(days=30*i)
    meses_passados.append({
        'month': data_mes.strftime('%Y-%m'),
        'valor': consumo_medio_real,
        'tipo': 'Consumo Médio Atual'
    })
```

Figure 10. Detecção de alertas e preparo dos dados históricos

Na Figura 11, o gráfico é efetivamente criado. O sistema combina os dados históricos e as previsões em um Dataframe, ocasionando a visualização utilizando o plotly.express.line() e adicionando elementos gráficos como:

- Linha Azul: Consumo médio atual;
- Linha verde: Previsão de consumo;
- Marcador verde: Estoque atual;
- Linha Vermelha: Estoque projetado.

```python
# Adicionar dados de previsão
dados_previsao = []
for _, row in previsao_df.iterrows():
    dados_previsao.append({
        'month': row['month'].strftime('%Y-%m'),
        'valor': row['previsao_consumo'],
        'tipo': 'Previsão'
    })

# Combinar dados
comparativo_data = meses_passados + dados_previsao
comparativo = pd.DataFrame(comparativo_data)
comparativo['month'] = pd.to_datetime(comparativo['month'])

# Traduzir nome da peça para o gráfico
selected_piece_traduzido = traduzir_peca(selected_piece)

# Gráfico baseado no estoque atual
fig = px.line(comparativo, x="month", y="valor", color="tipo",
              title=f"Consumo Médio Atual vs Previsão - {selected_piece_traduzido}",
              labels={"valor": "Quantidade (unid/mês)", "month": "Mês", "tipo": "Tipo"})
fig.update_traces(mode="lines+markers")

# Adicionar linha de estoque atual e projetado
if not previsao_df.empty:
    # Ponto atual do estoque
    fig.add_scatter(x=[data_atual], y=[estoque_atual],
                    mode='markers', name='Estoque Atual',
                    marker=dict(size=12, color='green', symbol='diamond'))

    # Linha de estoque projetado
    fig.add_scatter(x=previsao_df['month'], y=previsao_df['estoque_projetado'],
                    mode='lines+markers', name='Estoque Projetado',
                    line=dict(dash='dash', color='red'))
```

Figure 11: Montagem do gráfico preditivo Plotly

Finalizando as etapas, a figura Plotly é convertida em JSON e enviada ao template de arquivo HTML. Aqui é renderizado o gráfico e marcando a integração final entre a IA e a interface visual do sistema. Confira na imagem abaixo.

```
    # Linha de estoque projetado
    fig.add_scatter(x=previsao_df['month'], y=previsao_df['estoque_projetado'],
                    mode='lines+markers', name='Estoque Projetado',
                    line=dict(dash='dash', color='red'))

    # Adicionar linha de consumo médio como referência
    fig.add_hline(y=consumo_medio_real, line_dash="dot",
                  annotation_text=f"Consumo Médio: {consumo_medio_real:.1f} unid/mês",
                  line_color="orange")

grafico_json = pio.to_json(fig)

# Converter previsão em lista de dicionários
previsoes = previsao_df.to_dict(orient="records")

return render_template("predict.html",
                       selected_piece=selected_piece,
                       selected_piece_traduzido=selected_piece_traduzido,
                       previsoes=previsoes,
                       grafico_json=grafico_json,
                       metricas_peca=metricas_peca,
                       alertas=alertas)
```

Figura 12: Conversão do gráfico e renderização no template

O sistema respondeu de forma consistente às variações de consumo e gerou alertas preventivos estável. A interface interativa contribuiu para facilitar a análise dos resultados, tornando a visualização dos indicadores mais clara e acessível ao usuário.

4.3 - Gerenciamento e Atualização dos Estoques em Tempo Real

Na Figura 13, é apresentada a interface de gerenciamento de estoque, onde o usuário pode registrar novas movimentações, editar quantidades disponíveis e acompanhar o status atual de cada peça em tempo real.

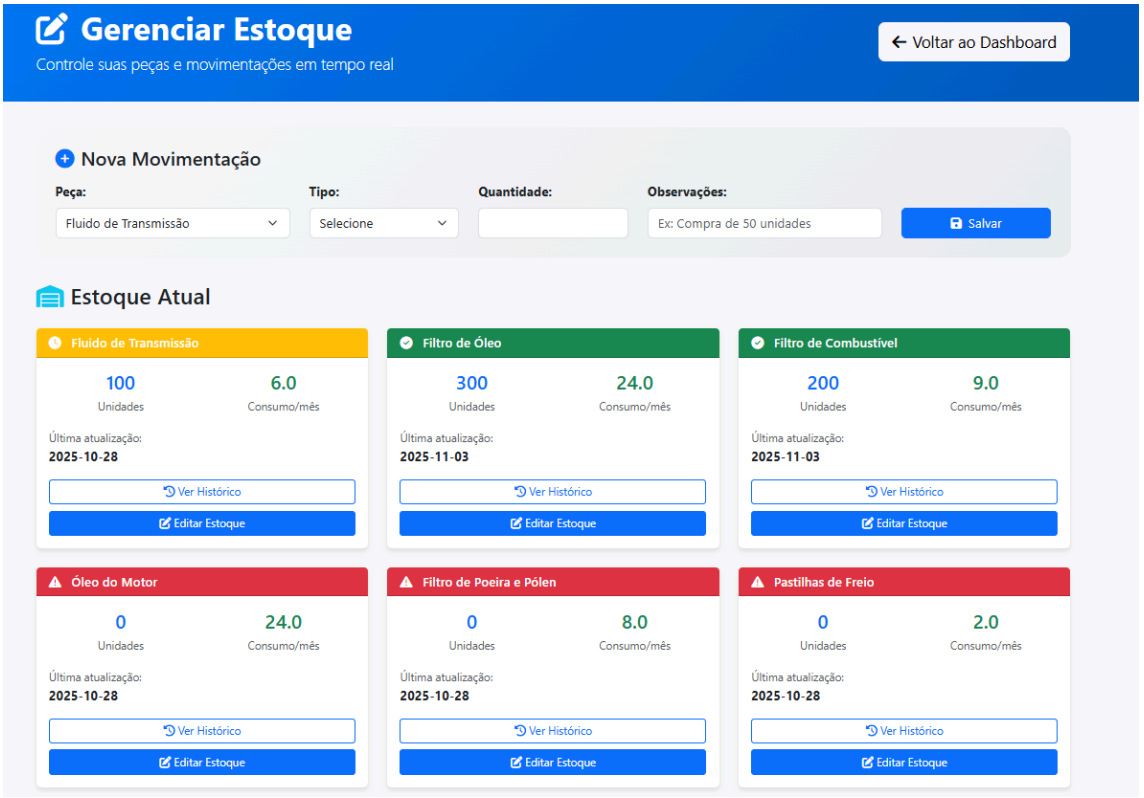

Figura 13: Interface gerenciamento de estoque

A interface presente na figura acima é a camada operacional do sistema, portanto o gestor realiza a atualização das informações diretamente pela aplicação web, sem necessidade de manipular manualmente os arquivos de dados.

- Cada card representa uma peça cadastrada e exibe:
- Quantidade atual: extraída do arquivo JSON, que reflete o estado em tempo real;
- Consumo médio mensal: calculado com base nos dados históricos do CSV;
- Data da última atualização;
- Status visual: indicado por cores distintas.

4.3.1 Atualização do estoque - Ações do usuário

Na sequência temos o trecho de código responsável pela atualização do estoque em tempo real, função que é acionada toda vez que o gestor realiza uma movimentação na interface da aplicação, como registrar uma entrada ou saída de peças.

Temos a presença de um método para atualizar estoques, responsável pela lógica necessária para refletir na base de dados operacional (arquivo JSON) as ações executadas pelo usuário.

```python
def atualizar_estoque(self, peca: str, quantidade: int, tipo_movimentacao: str, observacoes: str = ""):
    """
    Atualiza estoque de uma peça específica

    """
    if peca not in self.estoque_atual:
        # Inicializa peça se não existir
        self.estoque_atual[peca] = {
            'quantidade': 0,
            'ultima_atualizacao': datetime.now().strftime('%Y-%m-%d'),
            'consumo_medio': self._calcular_consumo_medio_historico(peca),
            'historico_movimentacoes': []
        }

    quantidade_anterior = self.estoque_atual[peca]['quantidade']

    # Calcular quantidade real baseada no tipo de movimentação
    if tipo_movimentacao == 'entrada':
        quantidade_real = quantidade  # Adiciona ao estoque
    elif tipo_movimentacao == 'saida':
        quantidade_real = -quantidade  # Subtrai do estoque
    else:
        quantidade_real = 0  # Não deveria acontecer com a validação

    nova_quantidade = max(0, quantidade_anterior + quantidade_real)

    # Registrar movimentação
    movimentacao = {
        'data': datetime.now().strftime('%Y-%m-%d %H:%M:%S'),
        'tipo': tipo_movimentacao,
        'quantidade': quantidade_real,
        'quantidade_anterior': quantidade_anterior,
        'quantidade_nova': nova_quantidade,
        'observacoes': observacoes
    }

    self.estoque_atual[peca]['historico_movimentacoes'].append(movimentacao)
    self.estoque_atual[peca]['quantidade'] = nova_quantidade
    self.estoque_atual[peca]['ultima_atualizacao'] = datetime.now().strftime('%Y-%m-%d')

    # Manter apenas últimas 50 movimentações
    if len(self.estoque_atual[peca]['historico_movimentacoes']) > 50:
        self.estoque_atual[peca]['historico_movimentacoes'] = \
            self.estoque_atual[peca]['historico_movimentacoes'][-50:]

    self._salvar_estoque_atual()
```

Figura 14: Atualização de estoque

4.3.2 Visualização das informações das peças

Na Figura 15, temos o trecho de código responsável por classificar cada peça de acordo com sua quantidade disponível e o consumo médio mensal, gerando um status visual exibido na interface. Ele calcula quantos dias o estoque atual conseguirá suprir a demanda, usando uma função para calcular os dias restantes. Com base nesse cálculo, o sistema define automaticamente níveis de criticidade e atribui cores distintas aos cards da tela

```python
def obter_status_estoque(self, peca: str) -> Dict:

    if peca not in self.estoque_atual:
        return {
            'status': 'Nao_Cadastrado',
            'cor': 'secondary',
            'mensagem': 'Peça não cadastrada no estoque',
            'dias_restantes': 0,
            'quantidade': 0
        }

    estoque_info = self.estoque_atual[peca]
    quantidade = estoque_info['quantidade']
    dias_restantes = self.calcular_dias_restantes(peca)

    # Classificar status
    if quantidade <= 0:
        status = 'Ruptura'
        cor = 'danger'
        mensagem = 'Estoque zerado - Compra urgente!'
    elif dias_restantes <= 7:
        status = 'Critico'
        cor = 'danger'
        mensagem = f'Acaba em {dias_restantes} dias - Compra imediata!'
    elif dias_restantes <= 15:
        status = 'Baixo'
        cor = 'warning'
        mensagem = f'Acaba em {dias_restantes} dias - Planeje compra'
    elif dias_restantes <= 30:
        status = 'Normal'
        cor = 'success'
        mensagem = f'Estoque saudável - {dias_restantes} dias restantes'
    else:
        status = 'Excesso'
        cor = 'info'
        mensagem = f'Estoque alto - {dias_restantes} dias restantes'

    return {
        'status': status,
        'cor': cor,
        'mensagem': mensagem,
        'dias_restantes': dias_restantes,
        'quantidade': quantidade,
        'consumo_medio': estoque_info['consumo_medio'],
        'ultima_atualizacao': estoque_info['ultima_atualizacao']
    }
```

Figura 15. Classificação status

4.3.3 Exibição geral de todas as peças na interface

O trecho de código abaixo pertencente a figura quinze, contém a função responsável por agrupar todas as peças que aparecem na tela da interface. Combinando os registros de peças do JSON (estoque atual) com as peças do CSV (base histórica), garantindo uma visão completa. Basicamente cada peça recebe status atual, quantidade e cor de status correspondente.

Antes de retornar dados, a função também realiza uma ordenação por prioridade garantindo que os itens críticos(ruptura ou baixo estoque) aparecem primeiro no painel, facilitando a identificação visual pelo gestor.

```python
def obter_todas_pecas_status(self) -> List[Dict]:
    """Obtém status de todas as peças cadastradas"""
    todas_pecas = list(self.estoque_atual.keys())
    # Adicionar peças do CSV que não estão no estoque atual
    pecas_csv = set(self.df_historico['part'].unique())
    pecas_faltantes = pecas_csv - set(todas_pecas)
    todas_pecas.extend(pecas_faltantes)

    status_pecas = []
    for peca in todas_pecas:
        status = self.obter_status_estoque(peca)
        status['peca'] = peca
        status_pecas.append(status)

    # Ordenar por prioridade (crítico primeiro)
    ordem_prioridade = {'Ruptura': 1, 'Critico': 2, 'Baixo': 3, 'Normal': 4, 'Excesso': 5, 'Nao_Cadastrado': 6}
    status_pecas.sort(key=lambda x: ordem_prioridade.get(x['status'], 7))

    return status_pecas
```

Figura 16. Exibição geral

4.4 Histórico de Movimentações de Peças

O recurso dessa seção permite ao gestor visualizar todas as operações realizadas de cada peça, como entradas, saídas, ajustes e observações registradas. Essa funcionalidade é exibida em forma de tabela dentro de um modal na interface conforme a imagem abaixo.

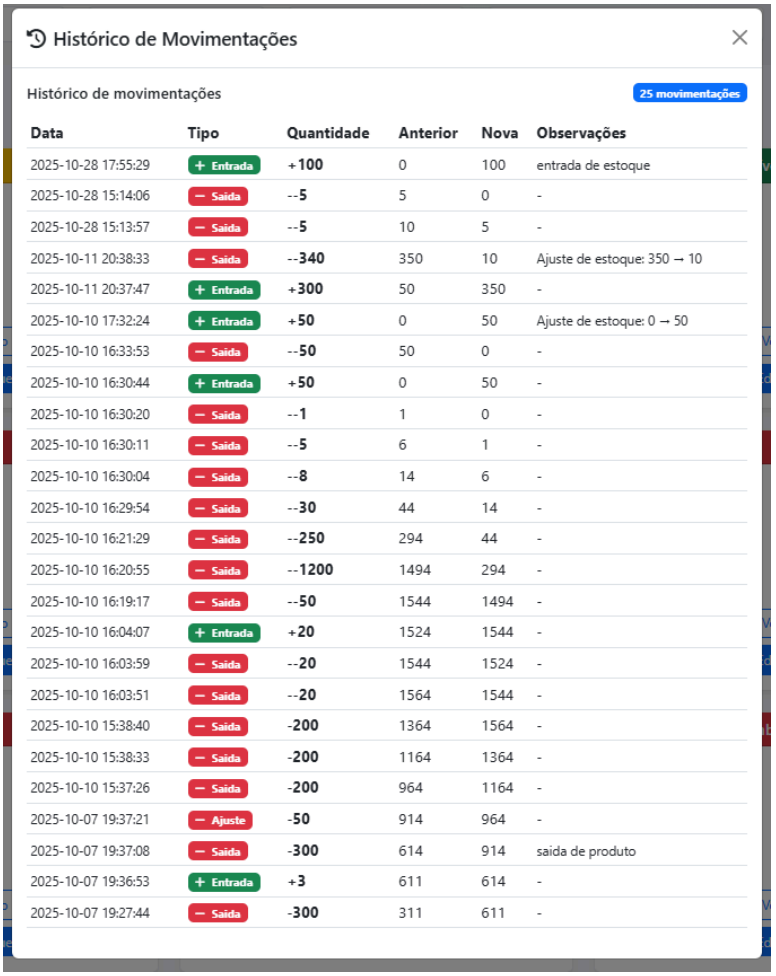

Figura 17: Interface gerenciamento de estoque

Todos esses dados são gerados automaticamente toda vez que for aplicado uma função que tem como tarefa atualizar o estoque, acionada no código, conforme na Figura 18.

```
# Registrar movimentação
movimentacao = {
    'data': datetime.now().strftime('%Y-%m-%d %H:%M:%S'),
    'tipo': tipo_movimentacao,
    'quantidade': quantidade_real,
    'quantidade_anterior': quantidade_anterior,
    'quantidade_nova': nova_quantidade,
    'observacoes': observacoes
}

self.estoque_atual[peca]['historico_movimentacoes'].append(movimentacao)
self.estoque_atual[peca]['quantidade'] = nova_quantidade
self.estoque_atual[peca]['ultima_atualizacao'] = datetime.now().strftime('%Y-%m-%d')
```

Figura 18: Registro de movimentação

5. Resultados e Discussão

O sistema desenvolvido integra recursos de análise, previsão e gerenciamento de estoque com foco em oficina mecânica, aplicado de forma web interativa. É presente a combinação da linguagem Python como framework Flask no back-end, as bibliotecas Pandas e Scikit-learn para processamento e modelagem de dados e para finalizar o Plotly para visualização dos gráficos.

A arquitetura possui uma atualização automática das métricas de estoque, sem a necessidade de alterar o arquivo CSV manualmente, refletindo em tempo real as movimentações de peças registradas.

Como objetivo geral o sistema tem o propósito de auxiliar o gestor de alguma oficina na tomada de decisão completamente baseada em dados, dessa maneira oferecendo indicadores sobre o consumo de peças, previsões e alertas preventivos da ruptura de estoque.

6. Considerações Finais

Este projeto final sobre previsão de estoques demonstra na prática como as técnicas de Inteligência Artificial contribuem para a gestão de materiais que nesse caso são direcionadas à oficinas mecânicas. A aplicação integrou dados históricos com informações operacionais, junção que gerou informações consistentes ao usuário final. Os resultados obtidos, especialmente por meio dos gráficos de previsão, se mostraram coerentes, auxiliando na identificação de cenários de normalidade até a ruptura.

Apesar da funcionalidade alcançada, o projeto apresenta algumas limitações. O modelo preditivo implementado, baseado em regressão linear, demonstrou capacidade na captura das tendências gerais do consumo, mas não passaram por uma avaliação quantitativa formal. Ausência de métricas de desempenho como MAE, RMSE ou MAPE, impedem a validação estatística da precisão das previsões. Como se trata de um modelo simples aplicado sobre séries históricas curtas, a capacidade preditiva fica restrita a padrões lineares, dessa forma, não contemplando variações sazonais ou comportamentos irregulares típicos de ambientes reais de manutenção automotiva.

Além disso, o sistema atualmente não está projetado para identificar variações de demanda. Como por exemplo, se alguma peça apresentar um aumento no consumo em alguns períodos específicos, se uma grande quantia de motorista realizar alguma revisão preventiva, o modelo não possui uma lógica sazonal, ele pode subestimar o consumo se algum existir esse cenário em específico. Consequentemente levando a ruptura ou subestimando os meses tranquilos, causando um excesso de estoque e aumento de custos. Esses comportamentos identificam que a regressão linear, embora seja funcional, possui limitações quando aplicada a dados completos e ambientes dinâmicos.

Atualmente a aplicação não está disponível para uso em mecânicas, as questões de melhorias serão continuadas em sequência a apresentação do projeto, dado que o

sistema pode ser de grande utilidade para as oficinas, sendo o foco principal do sistema a inclusão de uma tecnologia preditiva focada em estoques automobilísticos.

Os resultados não foram insuficientes, devido a ser um grande projeto o objetivo de ser algo preditivo foi concluído, gerando muita satisfação do que foi feito até aqui.

Como conclusão, considero este projeto de grande utilidade para a comunidade automobilística, principalmente para estabelecimentos de pequeno porte justamente pela falta de ferramentas de gerência que usam tecnologia, com o intuíto de facilitar toda a forma de organização e comunicação da parte logística de um determinado estabelecimento

7. Referências Bibliográficas

L. A. de Paula, *Explorando bibliotecas Python para visualização de campos meteorológicos do BRAMS*. Technical Report – Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2023.

L. Beraldi and E. Filho, *Impacto da tecnologia de informação na gestão de pequenas empresas*. Escola de Engenharia da USP, 2000.

V. A. Chaves Júnior, *Aplicação da Curva ABC na gestão de estoques de uma oficina mecânica*. Undergraduate Thesis — Instituto Federal do Espírito Santo (IFES), 2023. Available: https://repositorio.ifes.edu.br/handle/123456789/4101.

H. Y. Ching, *Gestão de estoques na cadeia de logística integrada – Supply chain*. 4th ed., Atlas, São Paulo, 2010.

A. R. Costa and L. S. Reis, "Visualização de Dados Interativa com Plotly em Python," *Revista de Engenharia e Tecnologia Aplicada*, vol. 9, no. 2, 2023.

A. Gerhardt, *Diagnóstico para o gerenciamento dos resíduos sólidos em oficina mecânica: estudo de caso em concessionária do município de Frederico Westphalen – RS*. Universidade Federal de Santa Maria (UFSM), 2010.

M. Gomes, *A Inteligência Artificial na Otimização da Gestão de Estoque*. Undergraduate Thesis – Faculdade Estácio de Sá, 2024.

J. Heizer and B. Render, *Operations Management*. Pearson, 2017.

S. E. Kimes, "The role of artificial intelligence in inventory management," *Journal of Hospitality Marketing & Management*, vol. 27, no. 2, pp. 204–221, 2018.

J. McCarthy, *What is Artificial Intelligence?*, 2007. Available: www.jmc.stanford.edu/articles/whatisai/whatisai.pdf.

H. Min, "Artificial intelligence in supply chain management: theory and applications," *International Journal of Logistics Research and Applications*, vol. 13, no. 1, 2010.

N. N. C. Menezes, *Introdução à Programação com Python: Algoritmos e lógica de programação para iniciantes*. São Paulo: Novatec, 2010.

V. N. Monteiro, G. A. Carvalho, A. F. S. Cruz, and J. V. C. Santos, "Integração de Python, Arduino, Machine Learning e Flask para criação de sistemas inteligentes de

automação e controle," *Apoena Revista Eletrônica*, vol. 6, pp. 218–223, 2023. Available: https://transformauj.com.br/apoena-revista-eletronica/.

F. C. Pereira Junior, *A influência da inteligência artificial no setor de negócios*, 2023.

PyData. "pandas: Time series / date functionality." Available: https://pandas.pydata.org/docs/user_guide/timeseries.html.

R. F. Quintanilha and G. C. S. Ruppert, "Estudo de Ferramentas para Ciência de Dados Aplicadas a Informações Médicas," *XXV Jornada de Iniciação Científica do CTI Renato Archer*, 2023.

S. J. Russell and P. Norvig, *Inteligência Artificial: uma abordagem moderna*. 2020.

V. L. B. Santos, D. A. Santos, J. M. B. Bianchi, and W. Salles, "A integração entre a Inteligência Artificial (IA) e a cadeia de suprimentos no contexto da Logística 4.0," FAETERJ – Faculdade de Educação Tecnológica do Estado do Rio de Janeiro, 2023. Available: https://www.aedb.br/seget/arquivos/artigos24/7735112.pdf.

Stefanini, "Benefícios da Inteligência Artificial: quais são os principais," 2018. Available: https://www.stefanini.com/pt-br/insights/artigos/beneficios-da-inteligencia-artificial-quais-sao-os-principais.

G. K. Taniwaki et al., "Desvendando os impactos da gestão estratégica no setor alimentício: benefícios e desafios," 2024.