CENTRO UNIVERSITÁRIO PARA O DESENVOLVIMENTO DO ALTO VALE DO ITAJAÍ - UNIDAVI

JACKSON FERNANDO WEEGE

SISTEMA DE ANÁLISE AUXILIAR PARA REPOSIÇÃO DE ESTOQUE

RIO DO SUL
2024

CENTRO UNIVERSITÁRIO PARA O DESENVOLVIMENTO DO ALTO VALE DO ITAJAÍ - UNIDAVI

JACKSON FERNANDO WEEGE

SISTEMA DE ANÁLISE AUXILIAR PARA REPOSIÇÃO DE ESTOQUE

Trabalho de Conclusão de Curso a ser apresentado ao curso de Sistemas da Informação, da Área das Ciências Naturais, da Computação e das Engenharias, do Centro Universitário para o Desenvolvimento do Alto Vale do Itajaí, como condição parcial para a obtenção do grau de Bacharel em Sistemas de Informação.

Prof. Orientador: Marco Aurélio Butzke

RIO DO SUL
2024

CENTRO UNIVERSITÁRIO PARA O DESENVOLVIMENTO DO ALTO VALE DO ITAJAÍ - UNIDAVI

JACKSON FERNANDO WEEGE

SISTEMA DE ANÁLISE AUXILIAR PARA REPOSIÇÃO DE ESTOQUE

Trabalho de Conclusão de Curso a ser apresentado ao curso de Sistemas da Informação, da Área das Ciências Naturais, da Computação e das Engenharias, do Centro Universitário para o Desenvolvimento do Alto Vale do Itajaí- UNIDAVI, a ser apreciado pela Banca Examinadora, formada por:

_____

Professor Orientador: Marco Aurélio Butzke

Banca Examinadora:

_____

Prof. Fernando Andrade Bastos

_____

Prof. Emanuela Valerio Jorge

Rio do Sul, 18 de novembro de 2024.

"Os números têm um impacto profundo na forma como vemos e compreendemos o mundo. Dominar a análise é transformar dados em decisões." (Hans Rosling).

Dedico este trabalho às pessoas cujo apoio foi indispensável em toda a minha jornada acadêmica: ao meu pai, à minha noiva e, especialmente, à minha mãe, cujo suporte diário tornou possível cada passo dessa caminhada.

AGRADECIMENTOS

Agradeço a todos que contribuíram para a qualidade do ensino durante minha trajetória no curso de Sistemas de Informação, em especial ao coordenador do curso, aos colegas, aos professores e, de maneira particular, ao meu orientador, Marco Aurélio Butzke, por seu apoio e orientação em todas as etapas do desenvolvimento deste trabalho.

RESUMO

O presente trabalho propõe o desenvolvimento de um protótipo para otimização do gerenciamento de estoque em uma plataforma e-commerce, com foco na análise de dados de vendas e previsão de demanda. O objetivo principal é oferecer uma visão clara sobre o uso de ferramentas e metodologias para automação de processos de reposição de estoque com conceitos de previsão e classificação de produtos conforme sua importância. Com o uso da linguagem de programação python e seus pacotes como pandas, numpy, e lifetimes, foi possível construir um sistema que organiza e analisa dados históricos de vendas, identificando padrões de compra e auxiliando na previsão de demanda futura. No protótipo foram utilizadas as técnicas de análise RFM (Recência, Frequência, Valor Monetário) e séries temporais, com objetivo de criar previsões que auxiliam na reposição eficiente do estoque. A aplicação foi desenvolvida com o framework streamlit para a visualização interativa dos dados e resultados das previsões. Nos testes realizados, a ferramenta apresentou alta precisão para produtos de alta demanda e venda contínua, reduzindo significativamente o tempo necessário para a análise de estoque e diminuindo as falhas para esses itens. Entretanto, para produtos de baixa demanda ou com grande variedade, a previsão ainda apresenta desafios devido à imprevisibilidade de vendas, um fator particularmente relevante no setor de papelaria, que possui uma vasta gama de itens. Os resultados indicam que o protótipo tem um potencial significativo para melhorar o controle de estoque e as operações em setores com vendas mais regulares e previsíveis. A aplicação pode ser adaptada para atender a diferentes nichos de mercado, especialmente onde há menor variedade de produtos, contribuindo para a otimização do tempo e a redução de falhas de estoque em empresas de comércio eletrônico.

Palavras-Chave: Gestão de estoque, previsão de demanda, E-commerce.

ABSTRACT

This study proposes the development of a prototype to optimize inventory management in an e-commerce platform, focusing on sales data analysis and demand forecasting. The main goal is to provide a clear understanding of the use of tools and methodologies for automating stock replenishment processes through forecasting concepts and product classification based on their importance. By utilizing the python programming language and its packages, such as Pandas, numpy, and lifetimes, a system was built to organize and analyze historical sales data, identify purchasing patterns, and assist in forecasting future demand. The prototype employed RFM analysis techniques (Recency, Frequency, Monetary Value) and time series analysis to create forecasts that support efficient stock replenishment. The application was developed using the streamlit framework for interactive visualization of data and forecast results. In the tests performed, the tool demonstrated high accuracy for high-demand and continuously sold products, significantly reducing the time required for inventory analysis and minimizing stockout events for these items. However, for low-demand products or those with high variability, forecasting still presents challenges due to sales unpredictability—a particularly relevant factor in the stationery sector, which encompasses a wide range of items. The results indicate that the prototype has significant potential to improve inventory control and operations in sectors with more regular and predictable sales. The application can be adapted to meet the needs of different market niches, particularly those with a smaller variety of products, contributing to time optimization and reduced stock failures in e-commerce businesses.

Keywords: Stock management, demand forecasting, E-commerce.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

APE	Absolute Percentage Error
API	Applicational Programming Interface
AWS	Amazon Web Services
CSS	Cascading Style Sheets
CSV	Comma-Separated Values
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
OTAN	Organização do Tratado do Atlântico Norte
PDF	Portable Document Format
REST	Representational State Transfer
RFM	Recência, Frequência e Valor Monetário
RMSE	Root Mean Squared Error
SKU	Stock Keeping Unit
TI	Tecnologia da Informação
URL	Uniform Resource Locator
WEB	Internet

SUMÁRIO

1. INTRODUÇÃO

Há centenas de anos o comércio é parte essencial do ecossistema econômico das sociedades, e com o avanço da tecnologia, o comércio continua evoluindo e alcançando novos patamares. O E-commerce hoje em dia já é um setor predominante, amplamente utilizado em todo o globo devido a sua acessibilidade, conveniência e praticidade.

A utilização de uma plataforma de vendas online apresenta diversos benefícios, como uma aproximação mais fácil do cliente, maior alcance de visibilidade, facilidade para criação de anúncios e campanhas, funcionamento ininterrupto, entre outros. Mas também se faz necessário um gerenciamento responsável de estoque, visto que o desleixo nesta função gera diversos problemas relacionados a capacidade de venda.

Um manejo inteligente do estoque significa a ausência de erros e produtos defasados, reposição adequada de produtos de acordo com seu volume de vendas, acompanhamento de nichos e tendências. A realização inadequada de qualquer um desses pontos resulta inevitavelmente no comprometimento de parte das vendas.

No entanto, este gerenciamento do estoque não é automático, é um trabalho de reconhecimento, que necessita de acompanhamento constante dos produtos e uma visão capaz de perceber as tendências do mercado. Realizar esse processo diariamente é necessário para o bom funcionamento do e-commerce, no entanto, consome tempo e recursos importantes, que poderiam ser mais bem aproveitados em outras funções.

Diante disso, esse trabalho visa desenvolver um serviço capaz de auxiliar no gerenciamento do estoque, efetuar uma análise de todos os dados de relação entre os produtos e as vendas, estipular os itens que precisam de reposição e ordená-los conforme prioridade, isso tudo de forma automatizada, assim, garantindo uma análise completa de todos os produtos e maior eficiência do processo.

1.1 PROBLEMA DE PESQUISA

Como otimizar a reposição de estoque de uma plataforma e-commerce?

1.2 OBJETIVOS

1.2.1 Geral

- Desenvolver um protótipo de sistema automatizado para gerenciamento de estoque em e-commerce, visando à análise de dados de vendas e previsão de demanda futura, com foco na eficiência do processo de abastecimento.

1.2.2 Específicos

- Identificar ferramentas e tecnologias disponíveis para automação e gerenciamento de estoque, explorando metodologias de machine learning;
- Mapear o processo atual de gerenciamento de estoque em e-commerce, identificando pontos críticos e fatores que influenciam a demanda, sazonalidade, e velocidade de reposição de produtos;
- Avaliar linguagem de programação e framework adequados para garantir funcionalidade e escalabilidade do protótipo;
- Realizar testes de validação do protótipo para garantir que o sistema atenda às necessidades do gerenciamento de estoque;

1.3 JUSTIFICATIVA

A crescente competitividade no mercado digital, especialmente no setor de e-commerce, exige soluções que otimizem a gestão de estoque e a previsão de vendas. Pequenos erros em projeções de compras podem levar a perdas financeiras significativas, seja por excesso de estoque ou pela falta de produtos para atender à demanda. Neste contexto, o desenvolvimento de um modelo preditivo baseado em dados históricos de vendas e técnicas avançadas de análise, como a modelagem RFM, é essencial para embasar decisões estratégicas de reposição de estoque.

Este trabalho justifica-se pela necessidade de integrar ciência de dados com práticas comerciais, oferecendo às empresas uma ferramenta acessível, confiável e escalável para melhorar o planejamento logístico. A aplicação, desenvolvida com a linguagem de programação python e a framework streamlit, visa simplificar o uso de modelos complexos, tornando-os utilizáveis mesmo por profissionais com pouca experiência em programação. Além disso, a inclusão de análises sazonais e a projeção ajustada fornecem insights mais precisos para o gerenciamento de produtos.

A inclusão desse sistema automatizado de análise deve representar um aumento da praticidade no exercício da reposição, eliminando a necessidade de fazer o reconhecimento e análise manual e demorada dos itens presentes no estoque, facilitando a reposição e proporcionando uma visão dos itens com maior potencial de venda.

1.4 CONTEXTUALIZAÇÃO DA EMPRESA

O e-commerce em questão é especializado no segmento de materiais de papelaria e tem apresentado crescimento constante. Originado como uma extensão online de uma loja física, o negócio expandiu significativamente nos últimos cinco anos, acumulando um catálogo com mais de sete mil itens únicos em estoque e atendendo a uma média de 100 pedidos por dia em todo o território nacional. Com uma equipe enxuta de menos de 10 colaboradores, o e-commerce enfrenta desafios operacionais que impactam diretamente a eficiência de suas operações.

Um dos principais desafios está relacionado à reposição de estoque, que atualmente é realizada de forma manual. A empresa lida com mais de 20 fornecedores diferentes, o que, combinado à ampla diversidade de produtos e à inconsistência nas vendas, torna o processo de análise manual para reposição trabalhoso e propenso a erros. Esse cenário frequentemente resulta em problemas como excesso de estoque de alguns itens e a falta de outros, além de consumir um tempo valioso da equipe.

O sistema atualmente utilizado pela empresa não oferece ferramentas de predição de vendas ou funcionalidades que automatizem a reposição de estoque. O recurso disponível para auxiliar nesse processo é um relatório de vendas simples, que informa apenas os itens vendidos, a quantidade de vendas e o saldo restante em estoque.

Nesse contexto, a implementação de uma ferramenta capaz de realizar análises automáticas e fornecer relatórios precisos sobre os itens e as quantidades necessárias para reposição representaria um avanço significativo. Tal solução traria maior eficiência ao processo, reduzindo falhas, otimizando o tempo dedicado a essas atividades e, consequentemente, melhorando a gestão operacional da empresa.

2. REFERENCIAL TEÓRICO

2.1. ENGENHARIA DE SOFTWARE

É a natureza tanto do ser humano quanto de organizações tentar aperfeiçoar seus serviços, aprimorar seus produtos, em busca de mais clientes e resultados financeiros. O desenvolvimento de programas de computador, assim como em qualquer outra área, iniciou-se de uma maneira mais primitiva e evoluiu ao longo dos anos até se tornar o processo complexo que conhecemos hoje como engenharia de software.

> Em qualquer caso, organizações sempre almejam entregar produtos satisfatórios para clientes e usuários (alta qualidade), dentro de custos e prazos compensadores (alta produtividade) e previsíveis (alta previsibilidade). Para atingir esses objetivos, as organizações têm que entender seus próprios processos de negócio, a fim de aperfeiçoá-los e obter resultados cada vez melhores. No caso de organizações produtoras de software, os processos de negócio são os processos relativos ao ciclo de vida do software (Paula Filho, 2019, p. 3).

A engenharia de software surgiu pela necessidade de otimizar o processo de desenvolvimento de sistemas, que por vezes era ineficiente e bagunçado. A utilização de metodologias já estabelecidas em outras engenharias trouxe diversos benefícios, tornando os processos mais eficientes, organizados, profissionais e menos custosos.

> Por muitos anos a Engenharia de Software foi utilizada com o objetivo de criar software de qualidade dentro dos custos e dos prazos estimados pelo cliente, evitando desperdícios de tempo, esforços, direções erradas e atrasos. A criação de software foi subestimada e realizada sem nenhuma metodologia, gerando erros em sistemas, como problemas de cálculos e perdas financeiras e de tempo. Nesse período, podemos dizer que houve a crise do software. Com isso, em 1967, a Organização do Tratado do Atlântico Norte (OTAN) designou o termo Engenharia de Software para adequar o processo de desenvolvimento de software com metodologias já utilizadas em outras Engenharias. Uma série de metodologias e técnicas passaram a ser utilizadas antes, durante e depois da criação de software. Dados históricos apontam que houve uma diminuição brutal nos problemas em software após a adoção dessas metodologias, fazendo com que a indústria de software pudesse entregar sistemas com maior qualidade, em menos tempo e com custos reduzidos de manutenção (Moraes; Zanin, 2020, p. 11).

A entrada da engenharia de software também dividiu o desenvolvimento em mais setores do que se tinha antes, e novos aspectos passaram a ser considerados na hora do planejamento de um novo software.

> A partir desse momento, os profissionais e as empresas de Tecnologia da Informação passaram a preocupar-se mais com os diversos setores que envolvem o desenvolvimento de sistemas, como análise de requisitos, análise de sistemas, desenvolvimento, testes e implantação. Neste contexto, derivaram diversas

> metodologias, métodos e processos para auxiliar e guiar o trabalho de cada um desses segmentos (Moraes; Zanin, 2020, p. 12).

A engenharia de software não trata apenas dos aspectos referentes especificamente a estrutura do sistema e processos de desenvolvimento, ela trata também dos recursos responsáveis por esse desenvolvimento, as pessoas.

> Engenharia de Software envolve planejamento. Planejamento diz respeito também a pessoas e cronograma de trabalho. Pessoas porque divide as responsabilidades, de forma individual ou coletiva. Cronograma porque conforme o planejamento é que os gestores têm a possibilidade de mensurar o tempo necessário para o desenvolvimento de cada projeto. Engenharia de Software envolve também a preocupação com a qualidade do produto. Qualidade, neste contexto, não se refere apenas à entrega de produtos em funcionamento, mas também ao atendimento das necessidades do cliente, e, por isso, a área da Engenharia de Software, que trata de engenharia de requisitos ou análise de requisitos, tem importância fundamental, na medida em que é por meio dela que as equipes de desenvolvimento recebem a expectativa do cliente sobre o produto que está sendo desenvolvido para buscar atendê-la (Morais; Zanin, 2020, p. 17).

A utilidade de ter processos e métodos bem alinhados se mostra até mesmo na parte prática do desenvolvimento. Uma equipe grande composta por muitas pessoas, poderia encontrar dificuldades em unificar as ideias e meios de cada pessoa sem uma liderança gerenciada pela engenharia de software.

> Além disso, na fase de desenvolvimento da programação em si, a Engenharia de Software se faz presente, uma vez que a escolha do processo ideal irá influenciar diretamente no trabalho cotidiano de todos os envolvidos, incluindo a programação. Esse fator ganha relevância ainda maior em times que trabalham em horários distintos ou locais geograficamente distribuídos (Morais; Zanin, 2020, p. 17).

A execução correta da engenharia de software dará frutos até mesmo após o desenvolvimento já estar concluído e o produto entregue ao cliente, já que um desenvolvimento organizado permitirá uma manutenção muito mais fácil e efetiva.

> Uma vez entregue o software para o cliente, a Engenharia de Software tem sua importância revelada no momento de realizar a manutenção nesse software. Isto porque, se o sistema tiver sido corretamente planejado, o código do sistema tende a estar mais limpo e com menos defeitos. Isso irá causar menos manutenção e facilitar as manutenções que precisam ser realizadas (Morais; Zanin, 2020, p. 17).

O ponto principal na elaboração de um software é atender a demanda para a qual esse software está sendo desenvolvido, seja um problema que precisa ser resolvido, ou alguma necessidade específica de um cliente.

> Um sistema computacional atende a diversas demandas quase que simultaneamente. Para isso, ele deve trazem entre suas funcionalidades características que atendam essas diferentes vertentes. Dentre elas, podemos destacar as necessidades de um cliente, de negócio (que traz restrições ao sistema conforme o negócio do cliente), de sistema e até de usuário, que será o responsável por fazer o uso do sistema quando este for finalizado (Morais; Zanin, 2020, p. 55).

Esses requisitos necessitam de ser devidamente documentados e arquivados, visto que serão basicamente o ponto guia durante o desenvolvimento do sistema. Todo o processo será realizado com o objetivo de atingir as demandas especificadas nos requisitos.

> A seleção das funcionalidades de um software se baseia em diversos aspectos, dentre eles temos diversos meios de se obter essas informações do cliente, para que o software atinja suas expectativas. É importante lembrarmos que um software é algo complexo, e dentro dessa complexidade ele traz milhares de minuciosos detalhes que devem estar bem estabelecidos pela equipe de desenvolvimento. Para isso, é interessante que haja a criação de uma documentação de requisitos do software, que trará todos os requisitos que o sistema deverá ter. Isso ajudará a nortear a equipe durante o ciclo de desenvolvimento (Morais; Zanin, 2020, p. 56).

Os requisitos, apesar de serem compostos pelos recursos solicitados pelos clientes, não são apenas isso. Os requisitos devem esclarecer também as mais diversas necessidades do sistema, incluindo as especificidades mais técnicas sobre o funcionamento e performance do software.

> Os requisitos do usuário trazem restrições e serviços que serão ofertados ao cliente. Eles expõem os requisitos funcionais e não funcionais, nos quais todas estas informações são passadas aos clientes por meio claro e simples, tendo em vista que não faz parte do contexto do cliente algumas informações técnicas. Já os requisitos do sistema constituem de todo um detalhamento acerca das funções e de suas respectivas limitações funcionais, já que são direcionados ao uso da equipe em si, podem trazer detalhes técnicos, desde que descrevam apenas o comportamento externo do software. Dessa forma, é notória a quantidade de requisitos que um sistema tem, assim como as diversas áreas as quais eles devem atender, ou seja, não se limitam apenas às funcionalidades gerais do software e da necessidade do cliente, devem atender a outros fatores, como as regras do negócio e até as necessidades do próprio sistema (Morais; Zanin, 2020, p. 57).

Apesar de muitos sistemas diferentes utilizarem recursos parecidos, e as vezes até iguais, como telas de login por exemplo, a documentação dos requisitos deve ser sempre refeita do zero quando se trabalha com um cliente novo, visto que mesmo que haja necessidade de um recurso que já foi desenvolvido em algum outro projeto, ainda sim a visão do cliente pode necessitar de algum aspecto mais singular, que não será satisfeito sem um trabalho personalizado.

> O documento de requisitos deve ser elaborado conforme as particularidades do software que está sendo desenvolvido. Jamais um documento servirá para outro

> software, pois cada sistema tem suas características próprias, assim como metodologias utilizadas, ferramentas, dentre outros detalhes. Porém, o que pode ser comum em vários projetos diferentes é a estrutura do documento (Morais; Zanin, 2020, p. 57).

Naturalmente os requisitos possuem uma hierarquia de prioridade, onde alguns requisitos são totalmente indispensáveis, outros podem afetar o bom desempenho do sistema caso não recebam a devida atenção, e alguns podem agregar, mas a sua ausência não afetaria o software.

> Diante de tamanha complexidade do software, devemos saber que a obtenção dos requisitos não termina após a concepção de sua documentação, inclusive outra informação que um documento de requisitos deve nos trazer é sobre a prioridade dos requisitos, que podem ser classificados como essencial, importante e desejável, ou, até mesmo, como prioridade baixa, média e alta. Essas definições dependem muito da nomenclatura que a equipe decidir utilizar. Porém, o objetivo é o mesmo, classificar o requisito conforme a prioridade de implementação e importância dele perante o sistema. Essa classificação é utilizada no gerenciamento do escopo das etapas do projeto e na definição das prioridades durante o desenvolvimento do sistema (Morais; Zanin, 2020, p. 62).

O conjunto de requisitos forma um aspecto extremamente importante do processo de engenharia de software: O escopo do projeto. É ele que delimita os limites de até onde o projeto deve ir, qual a magnitude necessária para o sistema que o projeto se propõe a construir.

> O escopo de um projeto de desenvolvimento de software possui várias definições. Sob o ponto de vista da análise de requisitos, podemos dizer que ele representa uma lista de funcionalidades que devem estar presentes no sistema de software. Quando dividimos o desenvolvimento do software em várias fases, e as fases em iterações, podemos denominar o escopo de desenvolvimento das iterações como uma baseline ou linha de base dos requisitos (Sbrocco; Macedo, 2012, p. 42).

Um dos aspectos mais adotados na engenharia de software hoje em dia são os métodos ágeis, um conceito de desenvolvimento que foca em entregar parcelas especificas do sistema a cada vez, ao invés de finalizar ele inteiro para somente então mostrar ao cliente. Isso permite realizar entregas mais rápidas e utilizar do feedback para manter o desenvolvimento sempre alinhado com as necessidades do cliente.

> As mudanças na economia, nos serviços concorrentes e nos novos produtos tem sido cada vez mais rápida e os negócios precisam responder com a mesma velocidade, para não perder essas novas oportunidades. Como o software está presente em todas as operações do negócio, é essencial que novos softwares sejam desenvolvidos rapidamente para não perder essas novas oportunidades e também responder às pressões da competitividade. Pensando em formas de desenvolver softwares com maior velocidade e mantendo a qualidade e a satisfação dos clientes, em 2001, Kent Beck e outros 16 desenvolvedores assinaram o “Manifesto para o desenvolvimento

> Ágil de Software", que deu origem aos processos de desenvolvimento para criar software útil rapidamente. Geralmente, são processos iterativos nos quais a especificação, o projeto, o desenvolvimento e o teste são intercalados. Usando esse método, o software não é desenvolvido e disponibilizado integralmente, e sim uma série de incrementos e a cada novo incremento uma nova funcionalidade do sistema (Vetorazzo, 2018, p. 12).

Para o bom funcionamento do projeto, os processos devem ser respeitados e seguidos de acordo com o que for estabelecido no planejamento. Os métodos ágeis trazem muitos benefícios ao desenvolvimento de sistemas, no entanto, nenhum processo trará resultado se não for executado conforme o planejamento.

> Para um gerenciamento efetivo, os padrões de qualidade são muito importantes e podem ser internacionais, nacionais, organizacionais ou de projeto. Os padrões de produto definem as características que os componentes devem ter, como, por exemplo, um estilo de programação. Já os padrões de processos vão definir como o processo de software deve ser instituído. A utilização desses padrões evita a repetição de erros e, também, auxilia na continuidade, caso a produção do produto seja continuada por outros membros da equipe (Vetorazzo, 2018, p. 75).

2.1.1. Cloud

Cloud computing é um novo segmento de tecnologia que apareceu a alguns anos atrás, e desde então tem ganhado cada vez mais força como uma nova alternativa de infraestrutura e serviços remotos.

> A computação em nuvem é uma inovação no setor da tecnologia da informação (TI), representando a solução para problemas de sobrecargas de computação e de necessidade de altos investimentos para montar uma infraestrutura completa nas organizações. Essa solução é apresentada pela oferta de software e de infraestrutura como serviços, com base na internet, trazendo mudanças significativas na maneira como o hardware pode ser projetado e adquirido.

A Cloud oferece além da estrutura, uma série de serviços e recursos extremamente úteis para diversos fins diferentes, cobrando apenas um valor referente a intensidade de utilização dos elementos disponibilizados na plataforma.

> A computação em nuvem (cloud computing) refere-se ao fornecimento de serviços de computação via internet, com o intuito de oferecer inovações mais rápidas, recursos flexíveis e economias de escala, os quais incluem servidores, armazenamento, banco de dados, rede e software, geralmente pagando-se apenas pelo uso (Silva; Soares; Serpa, 2020, p. 51).

Providenciar uma estrutura local para sustentar uma aplicação web não é simples e nem barato, especialmente para uma organização que esteja dando seus primeiros passos. Uma

alternativa viável para evitar a dificuldade e os custos que uma infraestrutura própria geraria, é a aquisição e utilização de serviços de cloud computing.

> Os assinantes se livram da responsabilidade relativa à administração do sistema, à manutenção, ao suporte de energia 24 × 7 e ao suporte de refrigeração. Essa é uma base para a economia de custos, porque os assinantes podem usar o serviço sem se preocupar com custos adicionais vindos da manutenção da infraestrutura. O provedor, por outro lado, pode oferecer o serviço a uma taxa nominal aos assinantes devido à grande base de clientes (Silva; Soares; Serpa, 2020, p. 234).

O benefício da utilização de uma infraestrutura Cloud não é apenas financeiro, mas também na questão de administração de todo o hardware físico, que pode ser complicado de se manejar corretamente. Com uma estrutura virtual não há risco de danificar equipamentos de hardware fisicamente.

De acordo com Silva, Soares e Serpa (2020) o gerenciamento da configuração computacional mesmo em ambientes menores como uma máquina local gera um trabalho excessivo, que é transferido para os fornecedores do serviço de nuvem, facilitando para os usuários se concentrarem em outros pontos.

Dessa forma, a utilização da infraestrutura cloud pode ser a opção ideal para empreendimentos que estejam começando, ou que não precisem necessariamente de uma infraestrutura própria, e que assim, possam focar em outros aspectos da organização.

No modelo de computação tradicional, os usuários precisam comprar ou adquirir recursos em quantidade significativa no início. Por outro lado, a computação em nuvem segue o modelo de serviço utilitário. Como o fornecedor organiza todos os recursos necessários nesse modelo, o investimento inicial dos assinantes para a aquisição de hardware ou software diminui drasticamente. Eles não precisam organizar nada além dos sistemas clientes para acessar os serviços em nuvem. (Silva; Soares; Serpa, 2020, p. 234).

2.1.2. Docker

Docker é uma ferramenta de auxílio para o desenvolvimento de softwares muito poderosa, oferecendo diversos recursos, mas tendo como seu principal foco a execução de containers.

A documentação oficial indica que o Docker (2024, n.p.) fornece um conjunto de ferramentas de desenvolvimento, serviços, conteúdo confiável e automações, utilizados individualmente ou em conjunto, para acelerar a entrega de aplicações seguras.

Um container é uma instância isolada dentro do sistema, que permite ao desenvolvedor rodar testes utilizando recursos sem a necessidade de instalação local.

> Um contêiner é um processo isolado rodando em uma máquina host que está separado de todos os outros processos que estão rodando nessa máquina host. Esse isolamento aproveita namespaces do kernel e cgroups, recursos que existem no Linux há muito tempo. O Docker torna essas capacidades acessíveis e fáceis de usar (Docker, 2024, n.p.).

Esses containers podem ser salvos e armazenados no formato de uma “imagem”, que pode ser acessada posteriormente para reproduzir o container.

> Um contêiner em execução utiliza um sistema de arquivos isolado. Esse sistema de arquivos isolado é fornecido por uma imagem, e a imagem deve conter tudo o que é necessário para executar uma aplicação - todas as dependências, configurações, scripts, binários, etc. A imagem também contém outras configurações para o contêiner, como variáveis de ambiente, um comando padrão para executar e outros metadados (Docker, 2024, n.p.).

A utilidade deste container está justamente em poder rodar a aplicação sem instalação local de dependências, mas além disso, no compartilhamento deste container com outras pessoas dentro da organização, sendo uma maneira simplificada e segura de se trabalhar em conjunto.

> O Docker oferece a capacidade de empacotar e executar uma aplicação em um ambiente isolado chamado contêiner. O isolamento e a segurança permitem que você execute muitos contêineres simultaneamente em um determinado host. Os contêineres são leves e contêm tudo o que é necessário para executar a aplicação, então você não precisa depender do que está instalado no host. Você pode compartilhar contêineres enquanto trabalha e ter certeza de que todos com quem você compartilha recebem o mesmo contêiner que funciona da mesma maneira (Docker, 2024, n.p.).

Por cima de toda essa utilidade, o Docker ainda é um sistema extremamente compatível e portável para diversos tipos de ambientes, tanto locais quanto virtuais, assim como definido na documentação.

> A plataforma baseada em contêineres do Docker permite cargas de trabalho altamente portáteis. Os contêineres do Docker podem ser executados no laptop local de um desenvolvedor, em máquinas físicas ou virtuais em um data center, em provedores de nuvem ou em uma mistura de ambientes (Docker, 2024, n.p.).

Essa estrutura de containers gerenciáveis do Docker também resulta em uma capacidade naturalmente escalável, o que o torna apropriado para o desenvolvimento de aplicações com tamanhos indefinidos ou inconstantes.

Docker (2024, n.p.) “A portabilidade e a natureza leve do Docker também facilitam o gerenciamento dinâmico de cargas de trabalho, escalando ou desativando aplicações e serviços conforme as necessidades do negócio, em quase tempo real”.

As aplicações baseadas em containers não precisam ser executadas singularmente, é possível executar aplicações compostas por um conjunto de containers.

O Docker utiliza uma arquitetura cliente-servidor. O cliente Docker se comunica com o daemon Docker, que realiza o trabalho pesado de construir, executar e distribuir seus contêineres Docker. O cliente Docker e o daemon podem rodar no mesmo sistema, ou você pode conectar um cliente Docker a um daemon Docker remoto. O cliente e o daemon Docker se comunicam usando uma API REST, através de sockets UNIX ou uma interface de rede. Outro cliente Docker é o Docker Compose, que permite trabalhar com aplicações compostas por um conjunto de contêineres.

2.1.3 Github

O GitHub é uma plataforma de armazenamento e compartilhamento de arquivos de projetos, sendo uma ótima ferramenta realizar o versionamento do seu software, gerenciar as alterações realizadas e trabalhar em conjunto com outras pessoas.

“O GitHub é uma plataforma baseada na nuvem onde você pode armazenar, compartilhar e trabalhar junto com outros para escrever código” (GitHub, 2024, n.p.).

A base para o funcionamento do GitHub é o Git, um software open source de controle de versão feito justamente para trabalho cooperativo.

Quando você faz upload de arquivos para o GitHub, você os armazena em um 'repositório Git'. Isso significa que, quando você faz alterações (ou 'commits') nos seus arquivos no GitHub, o Git começará automaticamente a rastrear e gerenciar suas alterações. Há muitas ações relacionadas ao Git que você pode realizar diretamente no GitHub, através do seu navegador, como criar um repositório Git, criar branches e fazer upload e edição de arquivos. No entanto, a maioria das pessoas trabalha em seus arquivos localmente (em seu próprio computador) e depois sincroniza continuamente essas alterações locais — e todos os dados relacionados ao Git — com o repositório 'remoto' central no GitHub. Há muitas ferramentas que você pode usar para fazer isso, como o GitHub Desktop (GitHub, 2024, n.p.).

2.2 APLICAÇAO WEB

Uma aplicação web, diferente de um site estático, precisa apresentar algum tipo de interação com o usuário, e não apenas exibir informações. Essa interação é feito a partir da junção do HTML e uma linguagem de programação, que vai manipular os dados a serem exibidos ao usuário.

> Uma aplicação Web é diferente de um site estático. No site estático, o conteúdo é um arquivo ou documento pré-formatado, em que, por exemplo, todo o conteúdo está nas marcações em HTML conhecidas como TAGs, e nenhuma informação é carregada a partir de outros documentos ou bases de dados. Já uma aplicação Web é caracterizada por construir dinamicamente o seu conteúdo, com dados provenientes de um banco de dados, a partir da interação do usuário com as páginas, via navegadores (Milleto; Bertagnolli, 2024, p. 28).

A linguagem de programação manipula os dados e os exibe ao usuário, essa comunicação é feita por meio de requisições enviadas do usuário ao servidor, e o retorno do servidor via HTML ao usuário. Essa interação ocorre com troca de informações em ambas as direções.

> Um servidor recebe uma solicitação de um cliente por meio de um navegador. O servidor procura pelo documento em um sistema de arquivos e o envia de volta ao navegador para ser exibido ao cliente. Os recursos do sistema (texto, imagem, vídeo e áudio) são interligados entre si por links, que é a forma usual de navegação em aplicações Web (Milleto; Bertagnolli, 2024, p. 28).

Essa interação ocorre com troca de informações em ambas as direções, por meio do protocolo HTTP.

De acordo com Duckett (2024, p. 58) “protocolo de Transferência de Hipertexto (HTTP) é um conjunto de regras que especificam como os navegadores devem solicitar as páginas e como os servidores devem formatar a resposta”.

2.2.1. HTML

HTML é uma linguagem de marcação que serve de base para o funcionamento de todas as páginas web.

“HTML, ou HyperText Markup Language, é uma linguagem de marcação utilizada para criar páginas acessadas a partir de um navegador. A característica principal dessas páginas é que elas utilizam hipertexto para viabilizar a navegação” (Milleto; Bertagnolli, 2014, p.72).

HTML, diferente de linguagens de programação, é composto por elementos estáticos,

que são interpretados pelos navegadores e convertidos nos textos devidamente formatados, e então são efetivamente exibidos para o usuário.

> A linguagem HTML é a base de criação de qualquer página para a Web, porém trata-se de uma linguagem que é composta por elementos estáticos e está fortemente focada na composição e formatação de documentos. HTML é a sigla para HyperText Markup Language, ou Linguagem de Formatação de Hipertexto, e trata-se de uma linguagem de marcação utilizada para criar páginas na Internet. (Oliveira; Zanneti, 2021, p. 9).

Esses elementos estáticos são demarcados por tags, e essas tags são combinadas de forma a construir a estrutura de hiper texto adequada para a exibição da página web de acordo com as necessidades do projeto.

> Todos os elementos que compõem uma página são posicionados por meio de comandos específicos da linguagem, denominados TAGs. Uma TAG é uma palavra especifica, definida em HTML, envolta por sinais de “menor que” (<) e “maior que” (>). De um modo geral, as TAGs aparecem em pares, uma indicando o início e a outra indicando o fim da marcação (Milleto; Bertagnolli, 2014, p.72).

2.2.2. CSS

Antes do CSS, a estilização das páginas era feita por meio de tags HTML específicas, o que significa que alterar o estilo de uma página requeria alterar o próprio HTML daquela página, sendo muito ineficiente se comparado ao CSS, que reúne as configurações de estilo em um lugar só.

> As folhas de estilo em cascata (ou CSS – Cascading Style Sheets) mudam a forma de organização das páginas. O HTML passa a ser utilizado somente como elemento para estruturar as páginas, e o CSS é utilizado na formatação da aparência das páginas. Com o CSS, é possível definir em um único local a formatação que será utilizada por cada TAG. Com isso, apenas um arquivo é alterado, sendo que a mudança é automaticamente propagada a todas as páginas que compõem o site (Miletto; Bertagnolli, 2014, p. 80).

Hoje, toda a estilização das páginas é realizada por meio do CSS. Nas palavras de Miletto e Bertagnolli (2014, p.80) “As folhas de estilo possibilitam criar estilos personalizados para títulos, listas, imagens, etc., além de permitirem a definição de cores, fontes, bordas, alinhamentos, entre outras características vinculadas à aparência das páginas Web”.

2.3. LINGUAGEM DE PROGRAMAÇAO

Linguagens de programação são o elemento base para a comunicação entre homem e máquina. Sem uma linguagem de programação, seria infinitamente mais difícil desenvolver softwares do que é hoje.

Santos, Saraiva e Fátima (2018) comparam as linguagens de programação com as línguas faladas no planeta terra, onde existem várias linguagens diferentes que cumprem um mesmo propósito de traduzir um algoritmo para uma linguagem que o computador entenda, mas de maneiras diferentes.

Essa necessidade de tradução se deve ao fato de que o computador executa o seu processamento de informações baseado em um fluxo de código binário, e não entende palavras humanas, e para os programadores, era muito complexo e difícil lidar diretamente com códigos em linguagem binaria.

> No início da era da computação, os programadores se viam obrigados a programar diretamente em linguagem de máquina, composta por inúmeras combinações de 0s e 1s, o chamado código binário. Como era muito difícil decorar a função que cada uma das combinações exercia no processador, criou-se então uma linguagem mais próxima do entendi- mento humano que foi denominada de Assembly (Alves, 2013, p. 16).

O Assembly foi o primeiro passo em direção à evolução das linguagens de programação, mas ainda era muito complexo e próximo demais de linguagem de baixo nível.

Nas Palavras de Alves (2013, p. 17) “Apesar do avanço, era difícil (e ainda é) programar em Assembly. Foram surgindo, então, as linguagens de programação de alto nível, como Pascal, BASIC e C”.

Com o avanço das linguagens, começamos a programar em alto nível, que oferece uma leitura e entendimento muito mais fácil aos desenvolvedores. No entanto, o computador não entende linguagem de alto nível, portanto é necessário traduzir essa linguagem de volta para baixo nível para que a máquina seja capaz de realizar os comandos solicitados. Isso pode ser feito por meio de interpretação ou compilação. A compilação é considerada mais rápida que a interpretação.

> Na interpretação, um processo empregado principalmente pela linguagem BASIC (Beginners All Purpose Symbolic Instruction Code) nos computadores das décadas de 1970 a 1990, o código fonte do programa é convertido em linguagem de máquina à medida que o programa é executado. Essa linguagem de máquina é a única que o computador realmente entende, já que ela é a combinação de 0s e 1s. A conversão é efetuada por um programa escrito totalmente em linguagem de máquina e denominado interpretador. Ele lê o código-fonte linha por linha e decodifica-o para que possa ser executado pelo microprocessador. Já em programas compilados, o código-fonte é

> inteiramente convertido em linguagem de máquina uma única vez. O software que efetua essa conversão é chamado de compilador. Normalmente o processo de compilação do código-fonte para código executável compreende dois passos, sendo uma geração de arquivos objetos e outro a ligação desses arquivos com as rotinas das bibliotecas de funções da linguagem. A principal vantagem dos programas compilados sobre os interpreta- dos é a velocidade de execução (Alves, 2013, p. 17).

Esses dois métodos são os mais comumente utilizados, mas não são os únicos que existem. Há outros métodos que são utilizados por diversas linguagens, e até métodos que são utilizados por linguagens específicas.

> Um terceiro método de execução de programas é denominado pseudocompilação. Nele, o compilador gera um código intermediário, com instruções em código de máquina, mas não resolve os endereços de chamada as rotinas da biblioteca de funções da linguagem. Durante a execução, um programa denominado Run-time é carregado na memória junto com o código compilado. Esse Run-time é responsável por determinar os endereços de memória das funções e procedimentos chamados pelo programa, durante o tempo de execução (Alves, 2013, p. 18).

2.3.1. Python

Python é uma das linguagens de programação mais difundidas atualmente, e a principal linguagem para diversos setores diferentes de manipulação de dados por conta das suas varias bibliotecas robustas com uma infinidade de recursos diferentes.

> Python é uma linguagem de programação poderosa e fácil de aprender. Ela oferece estruturas de dados de alto nível eficientes e uma abordagem simples, mas eficaz, para a programação orientada a objetos. A sintaxe elegante e a tipagem dinâmica de Python, junto com sua natureza interpretada, tornam-na uma linguagem ideal para scripts e desenvolvimento rápido de aplicações em diversas áreas e plataformas.
> O interpretador Python e sua extensa biblioteca padrão estão disponíveis gratuitamente, tanto em formato de código-fonte quanto em binário, para todas as principais plataformas, no site oficial de Python, https://www.python.org/, e podem ser distribuídos livremente. O mesmo site também oferece distribuições, módulos Python de terceiros, programas, ferramentas e documentação adicional.
> O interpretador Python pode ser facilmente estendido com novas funções e tipos de dados implementados em C ou C++ (ou outras linguagens que possam ser chamadas a partir de C). Python também é adequado como uma linguagem de extensão para aplicações personalizáveis. (Python, 2024, n.p.).

Python é uma linguagem de alto nível, interpretada, que combina a simplicidade e eficiência de linguagens tradicionais com uma abordagem moderna e acessível. Inspirada em modelos de linguagens amplamente difundidas ao longo de décadas, como C, Perl e BASIC, Python refina essas influências para oferecer uma experiência de programação mais fluida e produtiva. Sua popularidade crescente se deve à sua ampla aplicabilidade, facilidade de

aprendizado, robustas bibliotecas e uma comunidade ativa que continuamente contribui para seu avanço.

> o processo de execução de um programa em Python é semelhante ao empregado pela linguagem BASIC, que acompanhava os computadores pessoais do início da década de 1970 e fim da década de 1990. Ele consiste, basicamente, em um programa denominado interpretador de linguagem que varre o código-fonte linha a linha, convertendo cada uma delas em um código que pode ser efetivamente executado pelo processador. (Alves, 2021, p. 7).

2.3.1.1. Pandas

Pandas é uma biblioteca poderosa para manipulação e análise de dados em Python. Ela fornece estruturas de dados flexíveis, como o DataFrame, que é essencial para trabalhar com tabelas organizadas em linhas e colunas.

Um DataFrame é como uma planilha ou tabela SQL, permitindo fácil manipulação de dados, como filtragem, agrupamento, junção e cálculo. Com Pandas, é possível importar dados de diversas fontes, como arquivos CSV, Excel, bancos de dados e muito mais, tornando-a indispensável para tarefas em ciência de dados.

> Um Pandas DataFrames, como uma planilha, é composto de colunas e linhas. Cada coluna é um objeto pandas.Series. Um DataFrame é, de certa forma, parecido com um array NumPy bidimensional, com rótulos para colunas e índice. Mas diferente do array NumPy, um DataFrame pode conter diferentes tipos de dados. Você pode considerar um objeto pandas.Series como um array NumPy unidimensional com rótulos. O objeto pandas.Series, como um array NumPy, pode conter apenas um tipo de dado. O objeto pandas.Series pode usar muitos dos mesmos métodos vistos com os arrays, como min(), max(), mean() e medium(). (Behrman, 2023, p. 113).

2.3.1.2. Numpy

NumPy (Numerical Python) é uma biblioteca essencial para a computação científica em Python. Ela permite trabalhar com arrays multidimensionais de forma eficiente e oferece várias funções matemáticas para manipular dados numéricos.

É conhecida por realizar operações rápidas e otimizadas em grandes conjuntos de dados, usando vetorização. Além disso, serve como base para outras bibliotecas importantes, como Pandas e TensorFlow, sendo fundamental em áreas como ciência de dados e aprendizado de máquina.

> A biblioteca de terceiros NumPy é uma potência ao fazer ciência de dados em Python. Mesmo que você não use os arrays NumPy diretamente, irá encontrá-los porque são os blocos de construção para muitas outras bibliotecas. Bibliotecas como SciPy e Pandas se baseiam diretamente nos arrays NumPy. Esses arrays podem ser criados com muitas dimensões e tipos de dados. É possível adaptá-los para monitorar o consumo de memória controlando seu tipo de dado. Eles devem ser eficientes com grandes conjuntos de dados. (Behrman, 2023, p. 102).

2.3.1.3. Lifetimes

No contexto de análise preditiva e gestão de relacionamentos com clientes, a compreensão do valor do tempo de vida do cliente tem se tornado cada vez mais importante. Estimar a frequência e o valor das transações de um cliente ao longo de seu ciclo de vida é essencial para estratégias de retenção e alocação de recursos. O entendimento desse padrão e tendencia, pode garantir que as decisões tomadas sejam as corretas.

> Lifetimes is a powerful Python package for analyzing customer lifetime value (CLV) using a variety of models. It offers a simple and intuitive interface to model customer transactions, frequency, recency, and monetary value. Lifetimes includes popular models such as the Beta-Geometric/Negative Binomial Distribution (BG/NBD) and the Gamma-Gamma submodel, as well as extensions such as the Pareto/NBD and the Non-Contractual Bayesian model. Medium (2023, n.p.).

2.4. FRAMEWORK

Assim como em outras linguagens, o desenvolvimento em PHP também pode ser otimizado por meio da utilização de algum framework apropriado.

> Quando um desenvolvedor de sistemas decide trabalhar utilizando a linguagem de programação PHP, logo surge a dúvida sobre qual seria a melhor maneira de organizar os arquivos de todo o projeto. Para isso, é possível utilizar um framework que traga uma maneira mais automatizada de resolver a situação ou uma maneira própria de estruturar suas pastas e arquivos. Não existe um padrão de projetos ideal para ser utilizado, pois cada um tem suas finalidades e características específicas (Saraiva; Barreto, 2018, p. 255).

A utilização de um framework visa poupar o desenvolvedor de ter o trabalho de codificar funções e componentes comumente presentes em todo tipo de aplicação, fornecendo-as de maneira nativa. Freitas, Birnfield e Saraiva (2021, p. 48) afirmam que “framework é uma abstração que um e códigos comuns entre vários projetos de software, provendo uma funcionalidade genérica. Portanto, trata-se de uma forma mais simples de desenvolver aplicações pelo reuso de componentes”.

Dentre os recursos que os frameworks fornecem de maneira mais fácil ao desenvolvedor, estão aquelas que são indispensáveis para diversos tipos de aplicação.

O desenvolvimento Web de back-end consiste em inúmeras tarefas: proteger as application programming interfaces (APIs) de ataques de terceiros, autorizar usuários, facilitando a interação perfeita com bancos de dados, e rotear URLs para bancos de dados, com o objetivo de buscar informações solicitadas pelos clientes, entre outras. Os frameworks de back-end, também chamados de frameworks da Web no lado do servidor, tornam todas essas tarefas convenientes e descomplicadas para os desenvolvedores (Freitas; Birnfield; Saraiva, 2021, p. 71).

2.4.1. Streamlit

Streamlit é um framework poderoso e versátil que transforma o desenvolvimento de aplicações em Python em um processo mais intuitivo e eficiente. Com suas diversas ferramentas e funcionalidades, ele permite a criação de interfaces interativas de forma simplificada, sem a necessidade de conhecimentos avançados em desenvolvimento front-end. Sua facilidade de uso e capacidade de automatizar partes complexas do processo tornam o Streamlit uma escolha ideal para desenvolvedores que buscam praticidade e agilidade na construção de aplicações com python.

> Para construir um aplicativo da web simples no Streamlit, tudo o que um desenvolvedor precisa fazer é importar a biblioteca, criar uma função para renderizar a interface do usuário e, em seguida, executar o aplicativo com um único comando. Um exemplo disso é criar um aplicativo que permite ao usuário selecionar um arquivo CSV e exibir seu conteúdo em uma tabela interativa. Em poucas linhas de código, um desenvolvedor pode criar uma interface do usuário para seleção de arquivos, carregar o arquivo selecionado e exibir os dados em uma tabela interativa. AWARI (2023, n.p.).

3. METODOLOGIA DA PESQUISA

A metodologia adotada enquadra-se na categoria de pesquisa aplicada, visando gerar conhecimento para a solução de problemas específicos e práticos no contexto analisado. A modalidade de pesquisa escolhida é descritiva, com o objetivo de descrever as características de determinado fenômeno ou a relação entre variáveis. O trabalho buscou responder ao seguinte problema: Como otimizar a reposição do estoque em uma plataforma E-commerce? Para isso, levou-se em consideração 3 pontos principais: a eficiência do processo, a eficácia da seleção de itens, e a facilitação da leitura e análise das vendas.

O tipo de pesquisa realizado é um estudo de caso com análise qualitativa, proporcionando uma compreensão detalhada e aprofundada sobre o objeto de estudo. Esta abordagem permite explorar aspectos complexos e contextuais, fornecendo uma visão contextualizada do fenômeno investigado.

As técnicas de pesquisa empregadas incluem a aplicação de formulários, que permitem a coleta de dados primários e as necessidades diretamente dos participantes, e a análise documental, com foco nos relatórios de vendas do sistema. Estes relatórios fornecem dados quantitativos essenciais para a análise, enquanto os formulários capturam percepções e insights qualitativos dos participantes, enriquecendo a compreensão global do estudo.

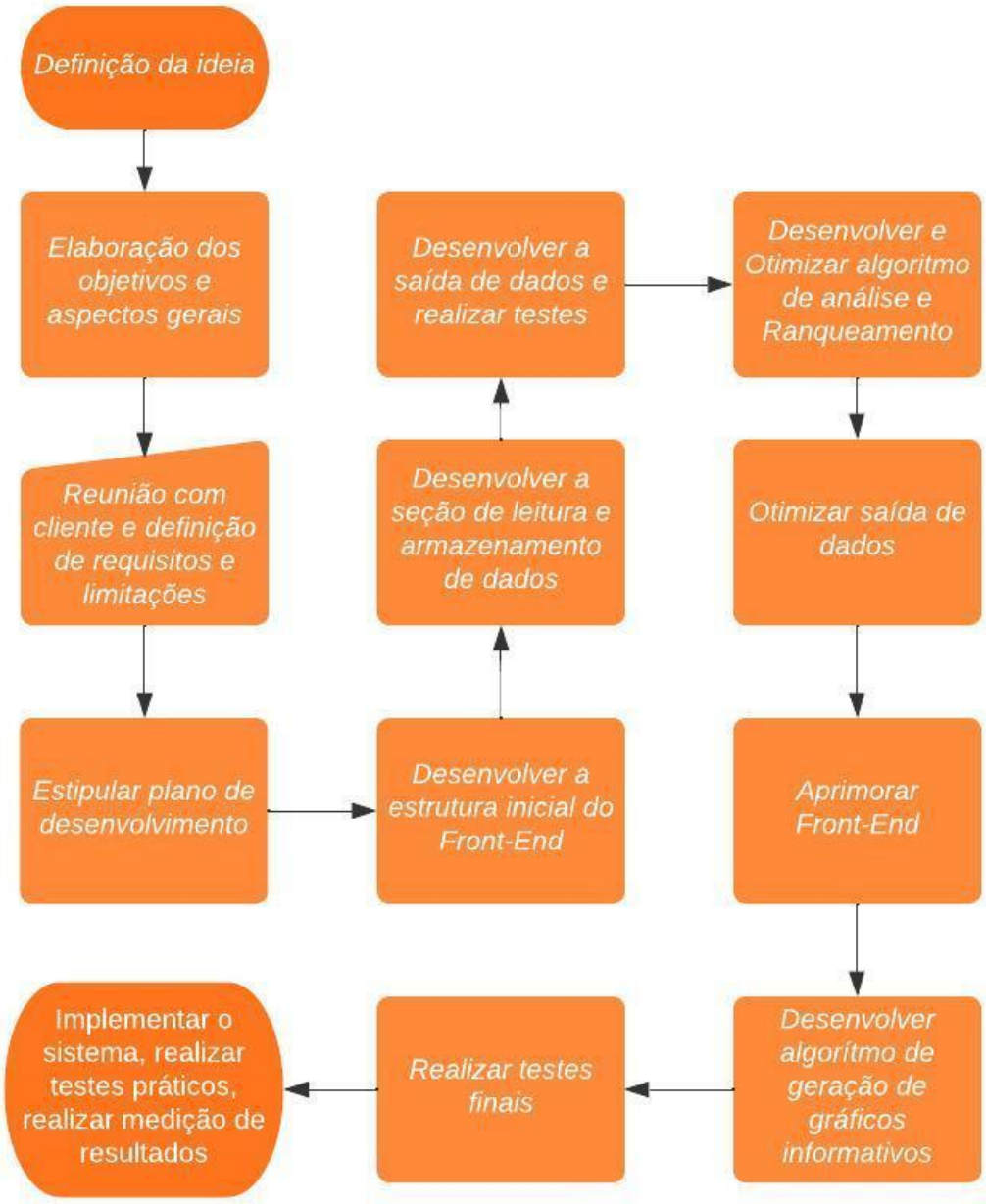
Com base nas informações obtidas do relatório de vendas, e as necessidades especificadas pelos administradores no formulário, foi feito um plano de desenvolvimento, definindo como classificar e priorizar os itens, levando em consideração aspectos como potencial, valor, disponibilidade e histórico dos itens.

Na etapa de coleta de dados, foram empregadas diversas ferramentas essenciais ao sucesso do projeto. O microsoft excel desempenhou um papel importante na organização e estruturação preliminar dos dados coletados, facilitando uma manipulação inicial das informações sobre transações e comportamento de clientes. Além disso, a coleta de dados transacionais, fundamentais para as análises, foi realizada a partir de registros de clientes e vendas da empresa, permitindo uma compreensão completa do histórico de compras.

Para o desenvolvimento e tratamento de dados, foi utilizado o ambiente de desenvolvimento integrado (IDE) visual studio code junto com a linguagem de programação python. Os pacotes pandas e numpy foram utilizadas para manipulação e análise eficiente dos dados. O pacote lifetimes foi aplicada especificamente para modelar e prever o comportamento de recompra dos clientes, e o streamlit foi utilizado para desenvolver a interface de visualização, possibilitando uma apresentação acessível e interativa dos dados e previsões.

O desenvolvimento do projeto foca, principalmente, na análise de dados transacionais para a previsão de vendas e o comportamento de recompra no contexto de e-commerce. As fases de obtenção, compreensão, preparação, modelagem, avaliação e visualização desses dados foram fundamentais para a criação de uma aplicação interativa e informativa, capaz de oferecer insights valiosos para o planejamento de vendas e estratégias de retenção de clientes conforme figura 1.

Figura 1 – Diagrama de etapas do desenvolvimento



Fonte: O Autor (2024).

4. APLICAÇÃO

A aplicação foi projetada para ser intuitiva e eficiente, oferecendo ferramentas capazes de processar dados históricos de vendas, realizar análises avançadas, e gerar previsões de demanda com base em técnicas modernas, como análise RFM e séries temporais. Levando isso em consideração, a linguagem de programação python foi escolhida por ser a mais apropriada para manipulação de dados. Além disso, o sistema utiliza uma interface interativa desenvolvida com streamlit, permitindo ao usuário explorar os dados e obter insights de maneira dinâmica.

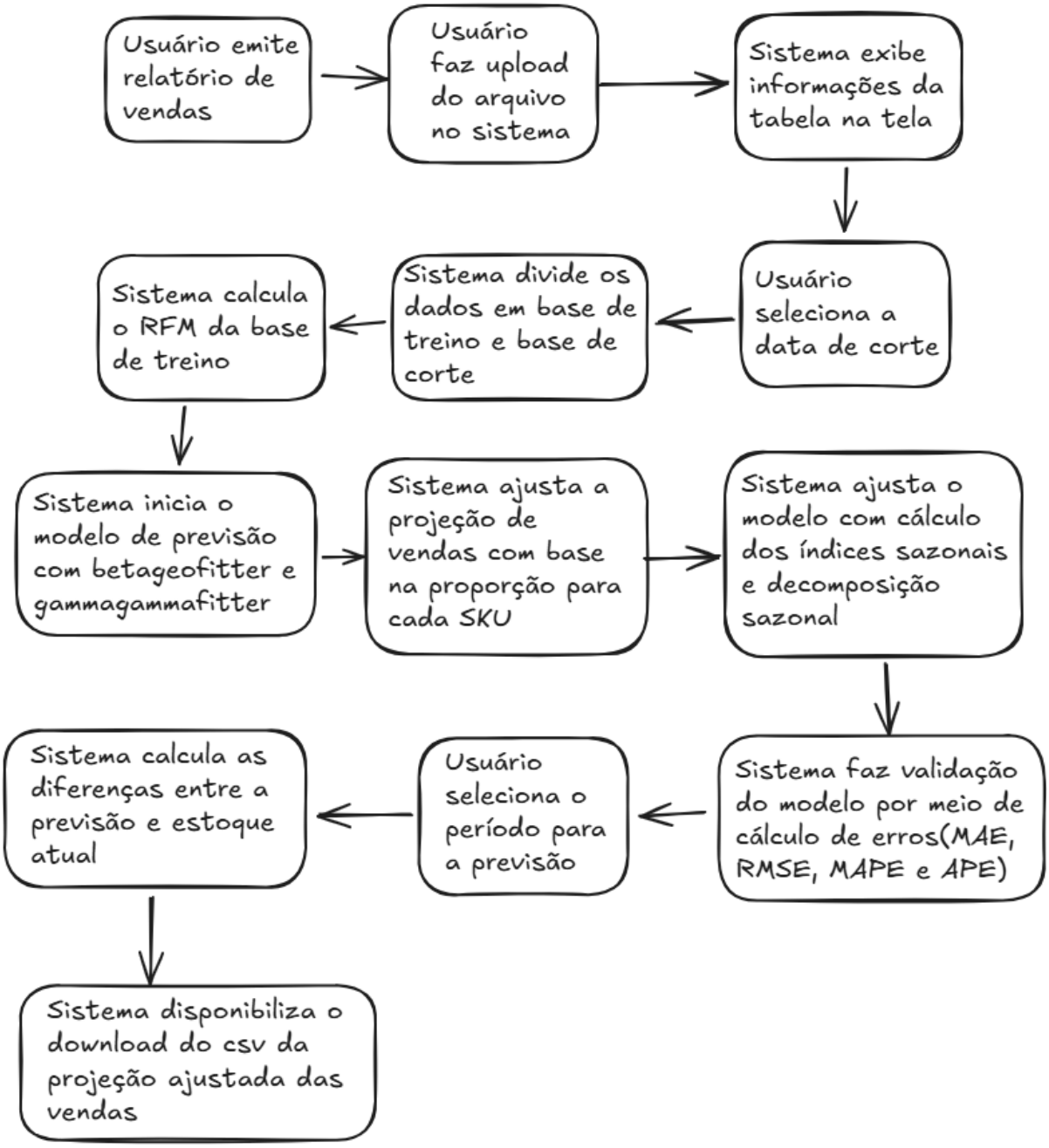
4.1 DIAGRAMA DE FUNCIONAMENTO DO SOFTWARE

A ferramenta desenvolvida tem como objetivo principal fornecer informações preditivas sobre as vendas, permitindo otimizar o processo de reposição de estoque. Para alcançar esse propósito, a aplicação segue uma sequência lógica única de processos, garantindo eficiência e clareza no fluxo de operações. Cada etapa foi projetada para complementar a anterior, resultando em um sistema coeso e eficiente.

Conforme a Figura 2, o diagrama descreve o funcionamento do sistema de previsão de vendas e ajuste de estoque. Primeiro, o usuário emite um relatório de vendas e faz o upload do arquivo no sistema. O sistema exibe as informações do arquivo, e o usuário define uma data de corte para dividir os dados em bases de treino e teste. Com os dados de treino, o sistema calcula os parâmetros de Recência, Frequência e Valor Monetário (RFM) e inicia o modelo de previsão usando BetaGeoFitter e GammaGammaFitter, estimando a frequência de compra e o valor médio das vendas para prever a demanda.

A projeção de vendas é ajustada conforme a proporção de vendas de cada SKU e, em seguida, é feita uma análise sazonal para identificar variações periódicas, aprimorando as previsões. O sistema valida o modelo com métricas de erro como MAE, RMSE, MAPE e APE, para avaliar a precisão das previsões. Em seguida, o usuário escolhe o período desejado para a projeção, e o sistema calcula a diferença entre as vendas previstas e o estoque atual. Por fim, o sistema gera um arquivo CSV com as projeções ajustadas, que pode ser baixado para uso na gestão de estoque.

Figura 2 – Diagrama de sequência de análise e processamento de dados.



Fonte: O Autor (2024).

4.2 CÓDIGO FONTE

Nesta seção de código, são importadas as bibliotecas e módulos essenciais para o desenvolvimento da aplicação.

O framework streamlit é utilizado para criar a interface visual interativa da ferramenta, enquanto pandas e numpy são fundamentais para a manipulação e análise dos dados. As funções BetaGeoFitter e GammaGammaFitter do pacote Lifetimes são carregadas para modelar o comportamento de recompra dos clientes, permitindo previsões com base em dados transacionais, como pode ser observado na figura 3.

Figura 3 - Configuração Inicial de Bibliotecas e Pacotes Necessários

app.py

```python
import streamlit as st
import pandas as pd
import numpy as np
from sklearn.metrics import mean_absolute_percentage_error, mean_absolute_error, mean_squared_error
from lifetimes import BetaGeoFitter, GammaGammaFitter
from lifetimes.utils import summary_data_from_transaction_data
from scipy import stats
from statsmodels.tsa.seasonal import seasonal_decompose

import warnings
warnings.filterwarnings('ignore')
```

Fonte: O autor (2024).

Conforme exibido na figura 4, este trecho de código define a função adjust_sales, que ajusta as vendas previstas para cada produto com base em índices sazonais específicos. A função recebe uma linha de dados e, em seguida, identifica o produto, a venda esperada e o índice sazonal associado no dicionário seasonal_indices.

Figura 4 - Ajuste de Vendas Previstas com Índices Sazonais

```python
def adjust_sales(row):  2 usages
    product = row['Nome']
    expected_sales = row['venda_produto_esperada']
    seasonal_info = seasonal_indices.get(product, {'index': 0, 'model': 'additive'})
    seasonal_index = seasonal_info['index']
    model_type = seasonal_info['model']
    if model_type == 'additive':
        adjusted_sales = expected_sales + seasonal_index
    else:
        adjusted_sales = expected_sales * seasonal_index
    return max(adjusted_sales, 0)

st.set_page_config(
    layout="wide",
    page_title='Previsão de Compras',
)
```

Fonte: O autor (2024).

A interface do streamlit permite carregar um arquivo de dados e configurar o modelo de previsão. Após o upload, os dados são exibidos na aba dados originais. Na aba Modelo de previsão, o usuário define a data de corte para separar treino e teste, ajustando também parâmetros como dias para validação e quantidade mínima de vendas. A análise RFM é gerada com base nos dados de treino para observar o comportamento dos produtos ao longo do período, conforme exibido na figura 5.

Figura 5 - Interface de Carregamento de Dados e Configuração do Modelo de Previsão

```
st.title('Modelo de Previsão de Compras')
arquivo = st.file_uploader('Baixe o Arquivo')
if arquivo is not None:
    st.session_state['df'] = pd.read_excel(arquivo)
    tabs = st.tabs(['Dados Originais', 'Modelo de Previsão', 'Previsão Futura'])
    with tabs[0]:
        st.dataframe(st.session_state['df'])
    with tabs[1]:
        st.subheader('Validação do Modelo na Base Informada:')
        st.write(
            f"Intervalo da Base: {st.session_state['df']['data'].min()} - {st.session_state['df']['data'].max()}"
        )
        datacorte = st.date_input('Selecione a Data de Corte')
        datacorte = pd.to_datetime(datacorte)
        t = st.slider('Número de dias para Validação do Modelo', value=10, min_value=3, max_value=30, step=1)
        quantidade_minima = st.slider('Quantidade Mínima', value=10, min_value=3, max_value=30, step=1)
        ## ajuste da base de treino e teste do modelo
        base_treino = st.session_state['df'][st.session_state['df']['data'] < datacorte]
        base_teste = st.session_state['df'][st.session_state['df']['data'] >= datacorte]
        base_teste = base_teste[base_teste['SKU'].isin(base_treino['SKU'])]
        ## geração da rotina de rfm
        rfm = summary_data_from_transaction_data(
            base_treino,
            customer_id_col='SKU',
            datetime_col='data',
            monetary_value_col='Qtd',
            observation_period_end=base_treino['data'].max()
        )
```

Fonte: O autor (2024).

Conforme exemplificado na figura 6, neste trecho é realizada a configuração do modelo RFM e as previsões de vendas. Primeiramente, o código filtra clientes com valor monetário positivo. Em seguida, o BetaGeoFitter estima o número esperado de compras durante o período de validação definido pelo usuário, e o GammaGammaFitter calcula o valor médio de vendas esperado por cliente. Por fim, o código multiplica a previsão de frequência de compras pelo valor médio, gerando as vendas esperadas para cada cliente ou SKU, ajudando a prever futuras demandas e ajustar a reposição de estoque com base nas métricas RFM.

Figura 6 - Configuração do Modelo RFM e Previsão de Vendas

```
        rfm = rfm[rfm['monetary_value'] > 0]
        bgf = BetaGeoFitter(penalizer_coef=0.05)
        bgf.fit(rfm['frequency'], rfm['recency'], rfm['T'])
        # previsão de dias para validação do modelo
        rfm['previsao_compra'] = bgf.conditional_expected_number_of_purchases_up_to_time(
            t, rfm['frequency'], rfm['recency'], rfm['T']
        )
        rfm['previsao_compra'] = rfm['previsao_compra'].clip(lower=0)
        rfm_final = rfm[rfm['frequency'] > 0]
        ggf = GammaGammaFitter(penalizer_coef=0.02)
        ggf.fit(rfm_final['frequency'], rfm_final['monetary_value'])
        rfm['media_vendas_esperada'] = ggf.conditional_expected_average_profit(
            rfm['frequency'],
            rfm['monetary_value']
        )
        rfm['media_vendas_esperada'] = rfm['media_vendas_esperada'].clip(lower=0)
        rfm['vendas_esperadas'] = rfm['previsao_compra'] * rfm['media_vendas_esperada']
        rfm['vendas_esperadas'] = rfm['vendas_esperadas'].clip(lower=0)
```

Fonte: O autor (2024).

Como mostrado na figura 7, este trecho de código organiza a projeção de vendas para produtos individuais. Ele agrupa os dados por SKU e Nome, calculando a proporção de cada produto no total de vendas por cliente, e utiliza a previsão de vendas gerada pelo modelo RFM para estimar a demanda esperada por produto (venda_produto_esperada). A tabela final projeta vendas agregadas por nome de produto, fornecendo insights detalhados para o gerenciamento de estoque.

Figura 7 - Projeção de Vendas por Produto com Ajustes de Proporção

```
        ### incluir nome do produto se possível
        cols = st.columns(3)
        if cols[0].toggle('Visualizar RFM'):
            cols[0].dataframe(rfm, use_container_width=True)
        produtos = base_treino.groupby(['SKU', 'Nome'])['Qtd'].sum().reset_index()
        produtos = produtos.merge(
            rfm[['vendas_esperadas', 'frequency', 'recency', 'monetary_value']],
            left_on='SKU', right_index=True, how='left'
        )
        quantidade_total_por_cliente = produtos.groupby('SKU')['Qtd'].transform('sum')
        produtos['proporcao_produto'] = produtos['Qtd'] / quantidade_total_por_cliente
        produtos['proporcao_produto'] = produtos['proporcao_produto'].clip(lower=0)
        produtos['venda_produto_esperada'] = produtos['vendas_esperadas'] * produtos['proporcao_produto']
        produtos['venda_produto_esperada'] = produtos['venda_produto_esperada'].clip(lower=0)
        projecao_venda_produtos = produtos.groupby('Nome')['venda_produto_esperada'].sum().reset_index()
        if cols[1].toggle('Visualizar a projeção de venda dos produto'):
            cols[1].dataframe(projecao_venda_produtos, use_container_width=True)
```

Fonte: O autor (2024).

Como exibido na figura 8, ajusta a projeção de vendas com base na sazonalidade, analisando vendas diárias de cada produto para identificar padrões sazonais. Ele define o tipo de modelo (aditivo ou multiplicativo) dependendo da presença de valores zero e realiza uma decomposição sazonal caso haja ao menos 90 dias de dados disponíveis. Em seguida, calcula um índice sazonal médio que ajusta a previsão de vendas do produto. Por fim, gera uma projeção ajustada por produto (vendas_esperadas_ajustadas) e permite a visualização da projeção final sazonalmente ajustada.

Figura 8 - Ajuste Sazonal das Projeções de Vendas

```
        # Ajuste Sazonalidade
        vendas_diarias = base_treino.groupby(['data', 'Nome'])['Qtd'].sum().reset_index()
        tabela_vendas_diarias = vendas_diarias.pivot(index='data', columns='Nome', values='Qtd').fillna(0)
        seasonal_indices = {}
        for produto in tabela_vendas_diarias.columns:
            serie = tabela_vendas_diarias[produto]
            has_zeros = (serie == 0).any()
            model_type = 'additive' if has_zeros else 'multiplicative'
            if len(serie.dropna()) >= 90:
                try:
                    decomposition = seasonal_decompose(serie, model=model_type, period=30)
                    seasonal = decomposition.seasonal
                    forecast_dates = pd.date_range(start=datacorte, periods=t)
                    forecast_seasonal = seasonal[seasonal.index.isin(forecast_dates)]
                    if not forecast_seasonal.empty:
                        seasonal_index = forecast_seasonal.mean()
                        seasonal_indices[produto] = {'index': seasonal_index, 'model': model_type}
                    else:
                        seasonal_indices[produto] = {'index': 0 if model_type == 'additive' else 1, 'model': model_type}
                except Exception as e:
                    print(f"Seasonal decomposition failed for product '{produto}': {e}")
                    seasonal_indices[produto] = {'index': 0 if model_type == 'additive' else 1, 'model': model_type}
            else:
                seasonal_indices[produto] = {'index': 0 if model_type == 'additive' else 1, 'model': model_type}
        projecao_venda_produtos['vendas_esperadas_ajustadas'] = projecao_venda_produtos.apply(adjust_sales, axis=1)
        projecao_final = projecao_venda_produtos[['Nome', 'vendas_esperadas_ajustadas']]
        if cols[2].toggle('Visualizar projecao final ajustada'):
            cols[2].dataframe(projecao_final, use_container_width=True)
```

Fonte: O autor (2024).

É então realizada a validação da projeção de vendas, comparando as vendas ajustadas previstas com as vendas atuais observadas no período de teste. Inicialmente, soma as vendas reais por produto e as integra na projeção ajustada. A seguir, calcula o erro absoluto percentual (APE) e a diferença absoluta de erro entre projeções e vendas reais. Além disso, são calculadas métricas de avaliação de precisão, incluindo MAE (erro médio absoluto), RMSE (raiz do erro quadrático médio), MAPE (erro percentual absoluto médio), e Mediana do APE. Por fim, apresenta uma tabela detalhada com essas métricas e dados de validação para produtos que atendem a um volume mínimo de vendas, como exemplificado na figura 9.

Figura 9 - Validação da Projeção de Vendas

```
        # Validação da Projeção
        vendas_atuais = base_teste.groupby('Nome')['Qtd'].sum().reset_index()
        vendas_atuais.rename(columns={'Qtd': 'vendas_atuais'}, inplace=True)
        validacao = projecao_final.merge(vendas_atuais, on='Nome', how='left')
        validacao['vendas_atuais'].fillna(0, inplace=True)
        validacao['APE'] = np.where(
            validacao['vendas_atuais'] == 0,
            np.nan,
            np.abs(
                (validacao['vendas_atuais'] - validacao['vendas_esperadas_ajustadas']) / validacao['vendas_atuais'])
        )
        validacao['Error'] = validacao['vendas_esperadas_ajustadas'] - validacao['vendas_atuais']
        mae = mean_absolute_error(validacao['vendas_atuais'], validacao['vendas_esperadas_ajustadas'])
        rmse = np.sqrt(mean_squared_error(validacao['vendas_atuais'], validacao['vendas_esperadas_ajustadas']))
        st.write(f"MAE: {mae:.2f}")
        st.write(f"RMSE: {rmse:.2f}")
        validacao_filtro = validacao[validacao['vendas_atuais'] >= quantidade_minima]
        mape = validacao_filtro['APE'].mean(skipna=True) * 100
        median_ape = validacao_filtro['APE'].median(skipna=True) * 100
        st.write(f"MAPE: {mape:.2f}%")
        st.write(f"Median APE: {median_ape:.2f}%")
        st.dataframe(
            validacao_filtro[['Nome', 'vendas_esperadas_ajustadas', 'vendas_atuais', 'APE']],
            use_container_width=True,
            hide_index=True
        )
```

Fonte: O autor (2024).

Como mostrado na figura 10, a ferramenta de previsão de compras é alimentada com os dados transacionais do e-commerce para calcular as previsões de vendas futuras. Primeiramente, é utilizada a abordagem de previsão de compras, que emprega o modelo BetaGeoFitter, responsável por calcular o número esperado de compras até um número determinado de dias, com base no comportamento histórico dos clientes, como especificado pelo slider de entrada. Em seguida, realiza-se a Análise RFM com o uso do modelo GammaGammaFitter, que calcula o valor monetário médio das vendas esperadas. Esse modelo combina a previsão de compras com o valor médio das vendas para estimar o total de vendas esperadas de cada produto.

Por fim, o Cálculo da Venda Esperada por Produto é realizado ao ajustar a previsão de compras conforme a participação de cada SKU no total de vendas do treinamento. Esse ajuste resulta em uma projeção mais precisa das vendas para cada item.

Figura 10 - Previsão de Compras e Análise RFM

```
147     with tabs[2]:
148         t = st.slider('Número de dias para Previsão', value=10, min_value=3, max_value=30, step=1)
149         rfm = summary_data_from_transaction_data(
150             st.session_state['df'],
151             customer_id_col='SKU',
152             datetime_col='data',
153             monetary_value_col='Qtd',
154             observation_period_end=base_treino['data'].max()
155         )
156         rfm = rfm[rfm['monetary_value'] > 0]
157         bgf = BetaGeoFitter(penalizer_coef=0.05)
158         bgf.fit(rfm['frequency'], rfm['recency'], rfm['T'])
159         rfm['previsao_compra'] = bgf.conditional_expected_number_of_purchases_up_to_time(
160             t, rfm['frequency'], rfm['recency'], rfm['T']
161         )
162         rfm['previsao_compra'] = rfm['previsao_compra'].clip(lower=0)
163         rfm_final = rfm[rfm['frequency'] > 0]
164         ggf = GammaGammaFitter(penalizer_coef=0.02)
165         ggf.fit(rfm_final['frequency'], rfm_final['monetary_value'])
166         rfm['media_vendas_esperada'] = ggf.conditional_expected_average_profit(
167             rfm['frequency'],
168             rfm['monetary_value']
169         )
170         rfm['media_vendas_esperada'] = rfm['media_vendas_esperada'].clip(lower=0)
171         rfm['vendas_esperadas'] = rfm['previsao_compra'] * rfm['media_vendas_esperada']
172         rfm['vendas_esperadas'] = rfm['vendas_esperadas'].clip(lower=0)
173         produtos = base_treino.groupby(['SKU', 'Nome'])['Qtd'].sum().reset_index()
174         produtos = produtos.merge(
175             rfm[['vendas_esperadas', 'frequency', 'recency', 'monetary_value']],
176             left_on='SKU', right_index=True, how='left'
177         )
```

Fonte: O autor (2024).

Conforme mostrado na figura 11, neste trecho o objetivo é ajustar as previsões de vendas com base no comportamento histórico e na sazonalidade dos produtos. Primeiro, é calculada a quantidade total por cliente e a proporção do produto em relação a essa quantidade. A proporção de cada produto é então usada para ajustar as vendas esperadas para cada item, levando em consideração sua participação no total de vendas. A soma das vendas esperadas de cada produto é calculada e armazenada em projeção_venda_produtos.

Em seguida, é realizada a análise de sazonalidade das vendas. Para isso, as vendas diárias por produto são agrupadas e organizadas em uma tabela que permite a análise da variação das vendas ao longo do tempo. A decomposição sazonal é realizada usando a função seasonal_decompose, que divide a série temporal em componentes sazonais. Dependendo da presença de valores zero nas séries de vendas, o modelo pode ser aditivo ou multiplicativo. Para cada produto, a média do índice sazonal durante o período de previsão é calculada. Se houver dados suficientes, o índice sazonal é utilizado para ajustar as previsões de vendas, considerando as flutuações sazonais.

O processo leva em conta a possibilidade de falhas na decomposição sazonal, garantindo que um índice padrão seja atribuído aos produtos que não podem ser decompostos adequadamente.

Figura 11 – Ajuste Sazonal e Projeção de Vendas

```
        quantidade_total_por_cliente = produtos.groupby('SKU')['Qtd'].transform('sum')
        produtos['proporcao_produto'] = produtos['Qtd'] / quantidade_total_por_cliente
        produtos['proporcao_produto'] = produtos['proporcao_produto'].clip(lower=0)
        produtos['venda_produto_esperada'] = produtos['vendas_esperadas'] * produtos['proporcao_produto']
        produtos['venda_produto_esperada'] = produtos['venda_produto_esperada'].clip(lower=0)
        projecao_venda_produtos = produtos.groupby('Nome')['venda_produto_esperada'].sum().reset_index()
        vendas_diarias = base_treino.groupby(['data', 'Nome'])['Qtd'].sum().reset_index()
        tabela_vendas_diarias = vendas_diarias.pivot(index='data', columns='Nome', values='Qtd').fillna(0)
        seasonal_indices = {}
        for produto in tabela_vendas_diarias.columns:
            serie = tabela_vendas_diarias[produto]
            has_zeros = (serie == 0).any()
            model_type = 'additive' if has_zeros else 'multiplicative'
            if len(serie.dropna()) >= 90:
                try:
                    decomposition = seasonal_decompose(serie, model=model_type, period=30)
                    seasonal = decomposition.seasonal
                    forecast_dates = pd.date_range(start=datacorte, periods=t)
                    forecast_seasonal = seasonal[seasonal.index.isin(forecast_dates)]
                    if not forecast_seasonal.empty:
                        seasonal_index = forecast_seasonal.mean()
                        seasonal_indices[produto] = {'index': seasonal_index, 'model': model_type}
                    else:
                        seasonal_indices[produto] = {'index': 0 if model_type == 'additive' else 1, 'model': model_type}
                except Exception as e:
                    print(f"Seasonal decomposition failed for product '{produto}': {e}")
                    seasonal_indices[produto] = {'index': 0 if model_type == 'additive' else 1, 'model': model_type}
            else:
                seasonal_indices[produto] = {'index': 0 if model_type == 'additive' else 1, 'model': model_type}
```

Fonte: O autor (2024).

As vendas esperadas ajustadas são calculadas por meio da aplicação do ajuste sazonal para cada produto. A projeção final é gerada, incluindo a quantidade esperada de vendas para cada item, e é combinada com as informações de estoque atual. A partir disso, é calculada a "Projeção Estoque", que indica a diferença entre o estoque atual e a quantidade projetada de vendas. O resultado é apresentado em uma tabela, ordenada de forma decrescente com base nas vendas esperadas ajustadas, permitindo uma análise detalhada da previsão de reposição de estoque para os produtos com maior demanda, conforme ilustrado na figura 12

Figura 12 - Projeção Final de Vendas e Estoque

```
        projecao_venda_produtos['vendas_esperadas_ajustadas'] = projecao_venda_produtos.apply(adjust_sales, axis=1)
        projecao_final = projecao_venda_produtos[['Nome', 'vendas_esperadas_ajustadas']]
        projecao_final['quantidade'] = round(projecao_final['vendas_esperadas_ajustadas'],0)
        projecao_final = projecao_final.merge(
            st.session_state['df'].groupby('Nome')['Estq Atual'].mean().reset_index(),
            on='Nome',
            how='left'
        )
        projecao_final['Estq Atual'] = round(projecao_final['Estq Atual'],0)
        projecao_final['Projecao Estoque'] = projecao_final['Estq Atual'] - projecao_final['quantidade']
        st.dataframe(
            projecao_final[projecao_final['vendas_esperadas_ajustadas'] > 0].sort_values(
                by='vendas_esperadas_ajustadas', ascending=False
            ),
            use_container_width=True
        )
```

Fonte: O autor (2024).

O DataFrame contendo as projeções de vendas ajustadas e os estoques é convertido em um arquivo CSV. Em seguida, é gerado um botão de download no Streamlit, permitindo que o usuário faça o download do arquivo CSV com a projeção ajustada. O arquivo contém os dados de vendas esperadas ajustadas, organizados em ordem decrescente, para análise e uso posterior, como ilustrado na figura 13.

Figura 13 - Exportação de Projeção Ajustada para CSV

```
        # Converter o DataFrame para CSV
        csv = projecao_final[projecao_final['vendas_esperadas_ajustadas'] > 0].sort_values(
            by='vendas_esperadas_ajustadas', ascending=False
        ).to_csv(index=False)

        # Botão de download
        st.download_button(
            label="Baixar Projeção Ajustada",
            data=csv,
            file_name="projecao_ajustada.csv",
            mime="text/csv"
        )
```

Fonte: O autor (2024).

4.3 TELAS DO SISTEMA

O sistema foi desenvolvido com foco na clareza e na facilidade de uso, apresentando uma interface simples e intuitiva. Ele se limita a poucas telas objetivas, organizadas de forma a atender às necessidades principais: entrada de dados, configuração das especificações e saída de informações preditivas. Essa abordagem visa proporcionar uma experiência eficiente para o usuário, permitindo que as funções essenciais sejam executadas de maneira rápida e direta.

Para a entrada de dados foi utilizado o relatório de vendas retirado diretamente do ERP da empresa, uma planilha contendo diversas informações relevantes e dados sobre as vendas de cada produto em cada dia presente no relatório, veja na figura 14.

Figura 14 - Relatório de vendas

SKU	Nome	Qtd	%*	Vlr Unit.	Vlr Total	%*	Curva	Custo Unit.	MKM	Margem %	Lucro	%*	Estq Atua	data
822	Marcador Para Quadro Branco V-Board Master PILOT R	56	13,93%	15,56	871,38	11,28%	A	0,00		100,00%	871,38	11,28%	1290	01/07/2024
JBO	Lápis de Cor Infinito Color'Peps Infinity 24 Cores MAPE	15	3,73%	34,81	522,10	6,76%	A	0,00		100,00%	522,10	6,76%	22	01/07/2024
TRILUX10SC	Caneta Esferográfica Trilux Com 10 Cores Faber-Caste	19	4,73%	20,89	396,90	5,14%	A	0,00		100,00%	396,90	5,14%	249	01/07/2024
CNX	Kit Mochila de Rodinhas + Lancheira Térmica + Estojo	1	0,25%	298,90	298,90	3,87%	A	0,00		100,00%	298,90	3,87%	0	01/07/2024
BCF	CANETINHA HIDROGRÁFICA BICOLOR FABER-CASTE	5	1,24%	49,90	249,50	3,23%	A	0,00		100,00%	249,50	3,23%	12	01/07/2024
MARE	Marcador Artístico Evoke Dual Marker BRW	1	0,25%	209,90	209,90	2,72%	A	0,00		100,00%	209,90	2,72%	15	01/07/2024
83	Refil Para Marcador V-Board Master (Quadro Branco) P	33	8,21%	6,29	207,52	2,69%	A	0,00		100,00%	207,52	2,69%	1036	01/07/2024
SMH	MOCHILA DE RODINHAS ESTRELAS	1	0,25%	207,04	207,04	2,68%	A	0,00		100,00%	207,04	2,68%	0	01/07/2024
CCI	Cartela de Carimbos Transparentes TILIBRA	9	2,24%	21,80	196,22	2,54%	A	0,00		100,00%	196,22	2,54%	182	01/07/2024
LILAS	LANCHEIRA TIRACOLO ACADÉMIE LISTRAS E POÁS	1	0,25%	173,90	173,90	2,25%	A	0,00		100,00%	173,90	2,25%	0	01/07/2024
CCSS	Caneca 3d Calopsita Cerâmica 300ml Broglio	3	0,75%	54,57	163,72	2,12%	A	0,00		100,00%	163,72	2,12%	28	01/07/2024
8756	Ponta p/ Marcador Quadro Branco Recarregável WBS-V	14	3,48%	11,36	159,08	2,06%	A	0,00		100,00%	159,08	2,06%	51	01/07/2024
JIN	Lancheira Térmica Escolar Lilica Ripilica Vem Ripilicar	1	0,25%	143,90	143,90	1,86%	A	0,00		100,00%	143,90	1,86%	0	01/07/2024
JBM	Caneta Visio Pen Canhoto Azul MAPED	6	1,49%	23,90	143,40	1,86%	A	0,00		100,00%	143,40	1,86%	168	01/07/2024
AJB	Estojo Box Playstation Astro Bot DERMIWIL	1	0,25%	136,99	136,99	1,77%	A	0,00		100,00%	136,99	1,77%	0	01/07/2024
PPK	Lancheira Sestini 2 Divisórias Lunch Crinkle Cyberpun	1	0,25%	135,22	135,22	1,75%	A	0,00		100,00%	135,22	1,75%	0	01/07/2024
DAJ	Pasta Suspensa Kraft Haste Plástica C/ Visor e Etiquet	50	12,44%	2,63	131,56	1,70%	A	0,00		100,00%	131,56	1,70%	340	01/07/2024
LZC	Garrafa Plástica Para Celular 500ml BRW	1	0,25%	112,90	112,90	1,46%	A	0,00		100,00%	112,90	1,46%	38	01/07/2024
DFO	Estojo Especial Triplo 3 Compartimentos Crinkle SEST	1	0,25%	106,94	106,94	1,38%	A	0,00		100,00%	106,94	1,38%	0	01/07/2024
TZU	CANECA 3D SHIH-TZU CERÂMICA 300ml BROGLIO	2	0,50%	52,90	105,80	1,37%	A	0,00		100,00%	105,80	1,37%	16	01/07/2024
KPD	Kit Carimbo Transparente Planner Completo TILIBRA	3	0,75%	35,19	105,57	1,37%	A	0,00		100,00%	105,57	1,37%	38	01/07/2024
JOH	Caneta Esferográfica Azul 0.7 com Pelúcia do Stitch MC	3	0,75%	34,90	104,70	1,36%	A	0,00		100,00%	104,70	1,36%	0	01/07/2024
JOK	Estojo Marcador Artístico Acrylic Paint Brush 12 Cores I	1	0,25%	98,90	98,90	1,28%	A	0,00		100,00%	98,90	1,28%	0	01/07/2024

Fonte: O Autor (2024).

O relatório de vendas contém as seguintes colunas de dados:

- **Qtd**: Refere-se à quantidade de unidades do item vendidas no dia, fornecendo uma visão clara do volume de vendas para cada produto.
- **%(quantidade)**: Indica a porcentagem dessa quantidade em relação ao total de vendas realizadas no dia, permitindo avaliar a participação de cada item nas vendas totais.
- **Vlr Unit.**: Exibe o valor unitário de cada item, refletindo o preço de venda por unidade.
- **Vlr Total**: Calcula o valor total das vendas de cada item, multiplicando o valor unitário pela quantidade vendida.
- **% (Valor Total)**: Representa a porcentagem do valor total das vendas de cada item em relação ao valor global das vendas realizadas no dia, oferecendo uma análise de contribuição para o faturamento.

- **Curva**: Classificação do item de acordo com a curva ABC, que categoriza os produtos com base em sua relevância para as vendas, permitindo identificar itens com maior e menor impacto no negócio.
- **Custo Unit.**: Informa o custo de cada item, embora este dado não esteja incluído na projeção de vendas.
- **MKM**: Coluna destinada a representar uma métrica adicional (MKM) específica, cuja descrição precisa ser fornecida no contexto do seu uso.
- **Margem**: Apresenta a margem de lucro do item, demonstrando a rentabilidade do produto após o custo e contribuindo para a análise financeira.

Detalhes na figura 15.

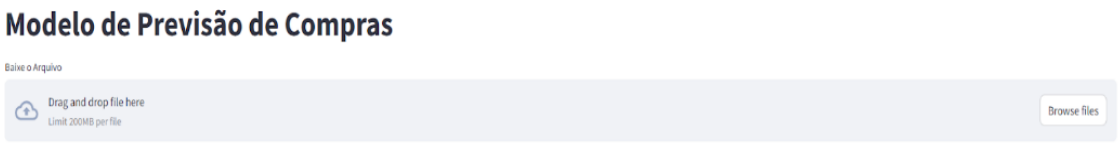
Figura 15 - colunas do relatório de vendas

Qtd	%*	Vlr Unit.	Vlr Total	%*	Curva	Custo Unit.	MKM	Margem %	Lucro	%*	Estq Atua	data
56	13,93%	15,56	871,38	11,28%	A	0,00		100,00%	871,38	11,28%	1290	01/07/2024

Fonte: O Autor (2024).

A primeira tela do sistema contém apenas o título da página e o campo para inserção dos dados, direcionando o usuário diretamente para a utilização da ferramenta. Veja na figura 16.

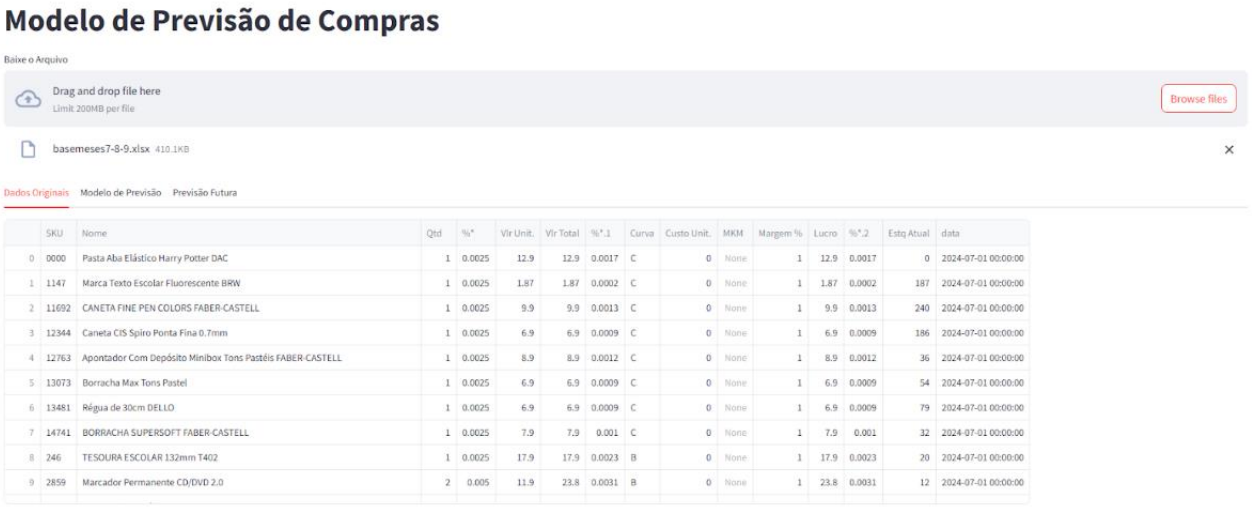
Figura 16 - Tela inicial, entrada de dados



Fonte: O Autor (2024).

Após a entrada de dados, carrega na tela a aba dados originais, exibindo uma tabela com os dados do relatório que foi inserido, além de outras duas abas, modelo de previsão e previsão futura, conforme mostra a figura 17.

Figura 17 - Tela inicial, após a entrada de dados

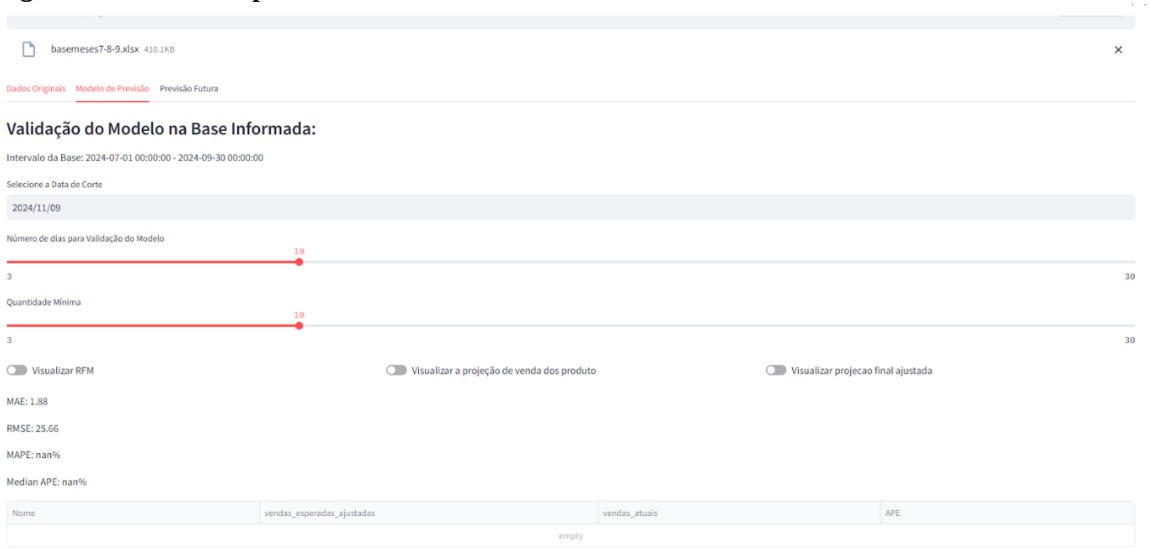


	SKU	Nome	Qtd	%*	Vlr Unit.	Vlr Total	%*.1	Curva	Custo Unit.	MKM	Margem %	Lucro	%*.2	Estq Atual	data
0	0000	Pasta Aba Elástico Harry Potter DAC	1	0.0025	12.9	12.9	0.0017	C	0	None	1	12.9	0.0017	0	2024-07-01 00:00:00
1	1147	Marca Texto Escolar Fluorescente BRW	1	0.0025	1.87	1.87	0.0002	C	0	None	1	1.87	0.0002	187	2024-07-01 00:00:00
2	11692	CANETA FINE PEN COLORS FABER-CASTELL	1	0.0025	9.9	9.9	0.0013	C	0	None	1	9.9	0.0013	240	2024-07-01 00:00:00
3	12344	Caneta CIS Spiro Ponta Fina 0.7mm	1	0.0025	6.9	6.9	0.0009	C	0	None	1	6.9	0.0009	186	2024-07-01 00:00:00
4	12763	Apontador Com Depósito Minibox Tons Pastéis FABER-CASTELL	1	0.0025	8.9	8.9	0.0012	C	0	None	1	8.9	0.0012	36	2024-07-01 00:00:00
5	13073	Borracha Max Tons Pastel	1	0.0025	6.9	6.9	0.0009	C	0	None	1	6.9	0.0009	54	2024-07-01 00:00:00
6	13481	Régua de 30cm DELLO	1	0.0025	6.9	6.9	0.0009	C	0	None	1	6.9	0.0009	79	2024-07-01 00:00:00
7	14741	BORRACHA SUPERSOFT FABER-CASTELL	1	0.0025	7.9	7.9	0.001	C	0	None	1	7.9	0.001	32	2024-07-01 00:00:00
8	246	TESOURA ESCOLAR 132mm T402	1	0.0025	17.9	17.9	0.0023	B	0	None	1	17.9	0.0023	20	2024-07-01 00:00:00
9	2859	Marcador Permanente CD/DVD 2.0	2	0.005	11.9	23.8	0.0031	B	0	None	1	23.8	0.0031	12	2024-07-01 00:00:00

Fonte: O Autor (2024).

A aba modelo de previsão apresenta as configurações para o funcionamento do modelo que devem ser especificadas de acordo com as necessidades do usuário. Apresenta também alguns dados do processamento dos dados, como o resultado da análise RFM realizada sobre os produtos, a projeção de vendas dos produtos e a projeção final ajustada. Por último, exibe abaixo algumas métricas relacionadas ao cálculo de erros do modelo. Ver na figura 18.

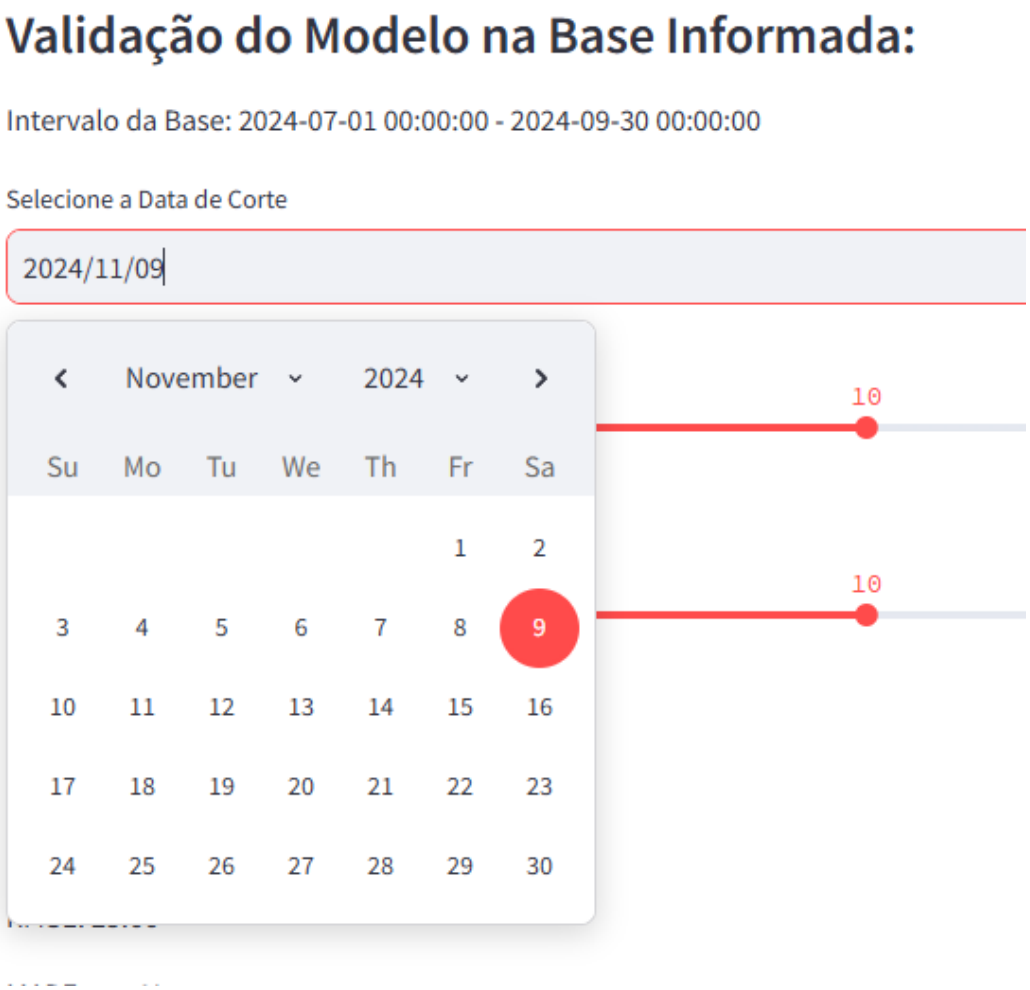
Figura 18 - modelo de previsão



Fonte: O Autor (2024)

Seletor da data de corte para o modelo de previsão. Define até qual dia, dentro do período contemplado no relatório de vendas, será considerado para o treino do modelo de previsão, conforme exibido na figura 19.

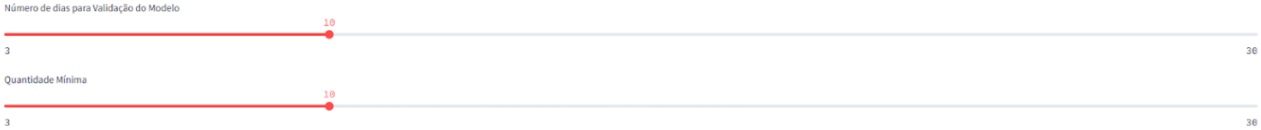
Figura 19 - modelo de previsão - seleção da data de corte



Fonte: O Autor (2024).

Seletor de número de dias para a validação do modelo, define o número de dias para a previsão, e o mesmo número utilizado para a validação. Seletor de quantidade mínima de vendas de cada produto para ser incluído no modelo de previsão, conforme exemplificado na figura 20.

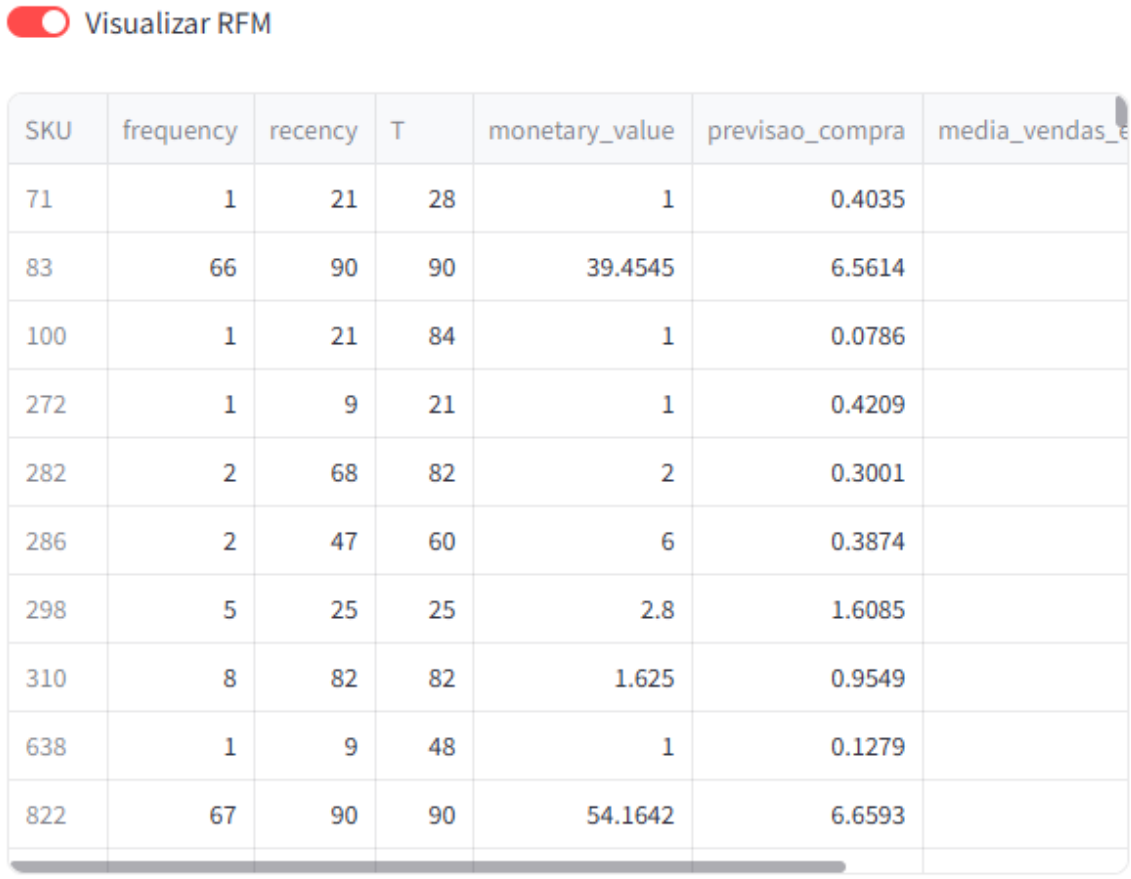
Figura 20 - modelo de previsão - número de dias para validação e quantidade mínima



Fonte: O Autor (2024).

Apresentação da análise RFM, feita sobre o relatório inserido na tela inicial. Apresenta dados como frequência, recência, valor monetário, previsão de compra, etc. Veja na figura 21.

Figura 21 - modelo de previsão - visualizar RFM



SKU	frequency	recency	T	monetary_value	previsao_compra	media_vendas_e
71	1	21	28	1	0.4035	
83	66	90	90	39.4545	6.5614	
100	1	21	84	1	0.0786	
272	1	9	21	1	0.4209	
282	2	68	82	2	0.3001	
286	2	47	60	6	0.3874	
298	5	25	25	2.8	1.6085	
310	8	82	82	1.625	0.9549	
638	1	9	48	1	0.1279	
822	67	90	90	54.1642	6.6593	

Fonte: O Autor (2024).

Apresentação da projeção de venda dos produtos, contendo o nome do produto e as vendas esperadas. Veja na figura 22

Figura 22 - modelo de previsão - projeção de venda dos produtos

Visualizar a projeção de venda dos produto

	Nome	venda_produto_esperada
0	Massa de EVA para Modelagem 10 Cores 50g MAKE+	0.5771
1	3 Refis Para Apagador de Quadro Branco 15-N Flip Top PILC	1.3116
2	APONTADOR COM BORRACHA E DEPÓSITO MIX FABER-CAS	1.2298
3	APOSTILA DE LETTERING BRW 26 FOLHAS	0
4	AQUARELA EM PASTILHA ACQUARELLI 36 CORES - GIOTTO	0
5	AQUARELA EM PASTILHA ESTOJO 12 CORES PINCEL FABER-	0
6	AQUARELA EM PASTILHA MINI 12 CORES - GIOTTO	0.5052
7	AQUARELA METÁLICA 12 CORES	0
8	Adaptador de Lápis com Encaixe no Dedo Color Joy 3un LE	1.2188
9	Adesivo Decorativo Cristal Animais Fofos UNISO	2.2548

Fonte: O Autor (2024).

Apresentação da projeção final após os ajustes de sazonalidade. Veja na figura 23.

Figura 23 - modelo de previsão - projeção final ajustada

Visualizar projecao final ajustada

	Nome	vendas_esperadas_ajustadas
0	Massa de EVA para Modelagem 10 Cores 50g MAKE+	0.5771
1	3 Refis Para Apagador de Quadro Branco 15-N Flip Top F	1.3116
2	APONTADOR COM BORRACHA E DEPÓSITO MIX FABER-C	1.2298
3	APOSTILA DE LETTERING BRW 26 FOLHAS	0
4	AQUARELA EM PASTILHA ACQUARELLI 36 CORES - GIOTT	0
5	AQUARELA EM PASTILHA ESTOJO 12 CORES PINCEL FAB	0
6	AQUARELA EM PASTILHA MINI 12 CORES - GIOTTO	0.5052
7	AQUARELA METÁLICA 12 CORES	0
8	Adaptador de Lápis com Encaixe no Dedo Color Joy 3un	1.2188
9	Adesivo Decorativo Cristal Animais Fofos UNISO	2.2548

Fonte: O Autor (2024).

Apresenta as métricas de erro na análise do sistema:

APE (Absolute Percentage Error - Erro Absoluto Percentual): O APE é uma medida de precisão que calcula a diferença percentual entre o valor real e o valor previsto. Ele é útil para entender o erro em termos relativos (porcentagem) em vez de valores absolutos. O APE é expresso como uma porcentagem e dá uma ideia de quão distante está a previsão em relação ao valor real, em termos percentuais.

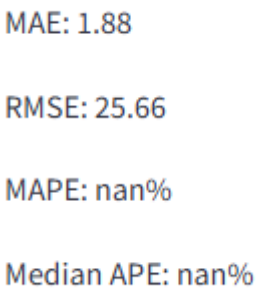
MAE (Mean Absolute Error - Erro Médio Absoluto): O MAE é uma medida de precisão que calcula a média dos erros absolutos entre os valores reais e previstos. Ele fornece uma ideia clara da magnitude média do erro, sem considerar a direção (se é maior ou menor). O MAE é útil para entender a precisão média do modelo.

RMSE (Root Mean Squared Error - Raiz do Erro Quadrático Médio): O RMSE é uma medida de precisão que calcula a raiz quadrada da média dos erros quadráticos entre os valores reais e previstos. Ele penaliza mais os erros grandes do que o MAE, pois os erros são elevados ao quadrado antes de serem somados.
O RMSE é frequentemente utilizado quando se quer dar maior peso aos erros mais significativos.

MAPE (Mean Absolute Percentage Error - Erro Percentual Absoluto Médio): O MAPE é a média do APE (erro percentual absoluto) para todas as observações. Ele fornece uma visão mais geral sobre o desempenho de um modelo, representando o erro médio em termos percentuais.
O MAPE é útil para medir o erro médio em porcentagem, o que facilita a interpretação do desempenho do modelo de previsão.

Essas métricas são amplamente utilizadas para avaliar o desempenho de modelos de previsão, pois fornecem diferentes perspectivas sobre o quão bem o modelo está se comportando em relação aos dados reais. Exemplificado na figura 24.

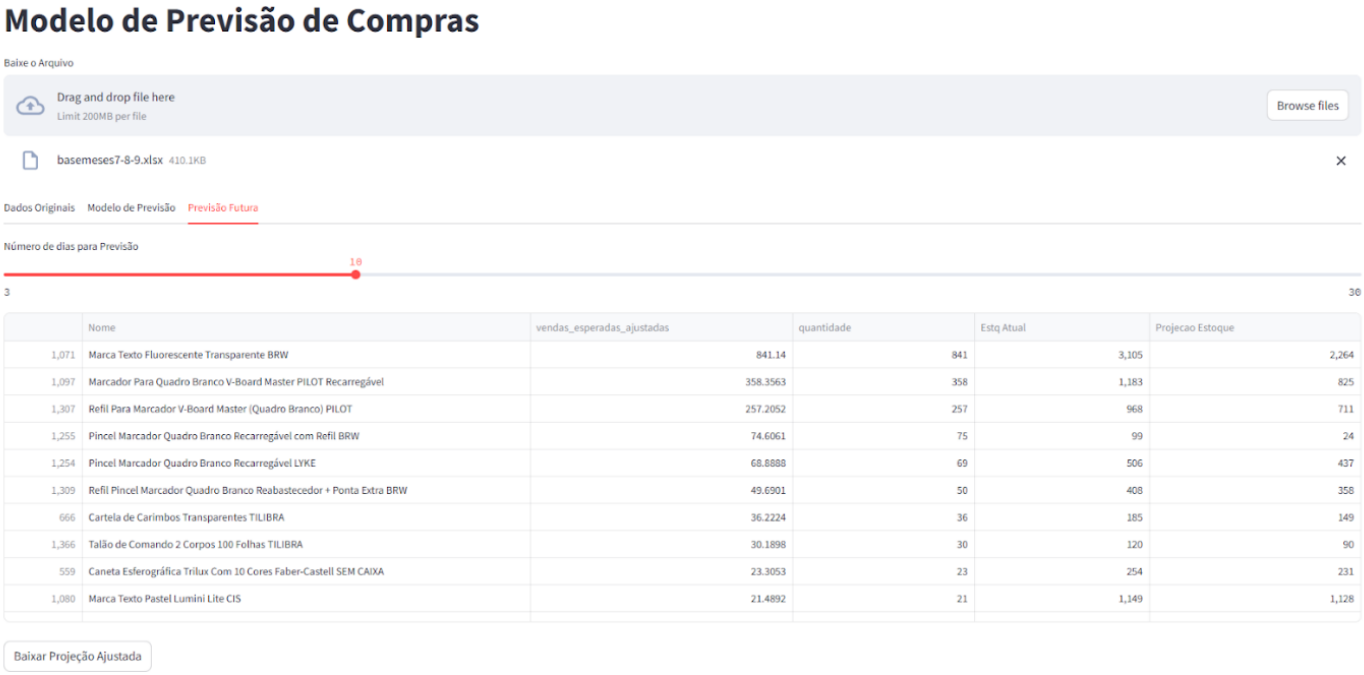
Figura 24 - modelo de previsão - métricas de erro do sistema



Fonte: O Autor (2024).

Conforme exibido na figura 25, tela final do sistema, apresenta o seletor para o número de dias que se deseja prever as vendas, o resultado da previsão futura, contendo o nome do produto, as vendas esperadas ajustadas e seu valor arredondado, o estoque atual e a projeção de estoque após essas vendas se concretizarem. há também um botão separado ao final da tela, para realizar o download da projeção ajustada.

Figura 25 - previsão futura



	Nome	vendas_esperadas_ajustadas	quantidade	Estq Atual	Projecao Estoque
1,071	Marca Texto Fluorescente Transparente BRW	841.14	841	3,105	2,264
1,097	Marcador Para Quadro Branco V-Board Master PILOT Recarregável	358.3563	358	1,183	825
1,307	Refil Para Marcador V-Board Master (Quadro Branco) PILOT	257.2052	257	968	711
1,255	Pincel Marcador Quadro Branco Recarregável com Refil BRW	74.6061	75	99	24
1,254	Pincel Marcador Quadro Branco Recarregável LYKE	68.8888	69	506	437
1,309	Refil Pincel Marcador Quadro Branco Reabastecedor + Ponta Extra BRW	49.6901	50	408	358
666	Cartela de Carimbos Transparentes TILIBRA	36.2224	36	185	149
1,366	Talão de Comando 2 Corpos 100 Folhas TILIBRA	30.1898	30	120	90
559	Caneta Esferográfica Trilux Com 10 Cores Faber-Castell SEM CAIXA	23.3053	23	254	231
1,080	Marca Texto Pastel Lumini Lite CIS	21.4892	21	1,149	1,128

Fonte: O Autor (2024).

4.3 AVALIAÇÃO DA APLICAÇÃO

O feedback foi coletado diretamente dos funcionários responsáveis pela reposição, que avaliaram o sistema com base nos resultados apresentados e na sua usabilidade prática. Essa interação permitiu ajustar algumas funcionalidades, aprimorando a interface e adaptando algumas métricas para uma visão mais precisa e alinhada com o contexto do e-commerce. Os funcionários destacaram que o sistema representa um avanço, mas também trouxeram sugestões importantes para aprimorar a priorização e facilitar a usabilidade, visando torná-lo uma ferramenta mais intuitiva e eficiente para o gerenciamento diário de estoque.

Os resultados dos testes mostram que o protótipo possui excelente desempenho na previsão de reposição para itens com alta demanda e volume significativo de vendas, oferecendo uma estimativa precisa para esses produtos. Com isso, a ferramenta consegue suprir a necessidade de antecipação de compras e otimização do controle de estoque para os itens mais vendidos. Entretanto, produtos com baixa demanda, vendas esporádicas, ou com um alto

número de variações representam um desafio para a previsão. No setor de papelaria, onde há uma vasta variedade de marcas e tipos de produtos, esses itens apresentam um comportamento de venda imprevisível, dificultando o acerto nas previsões e ocasionando divergências em relação às vendas reais.

A eficiência de tempo se mostrou um dos maiores benefícios da implementação do protótipo, reduzindo em cerca de 95% o tempo necessário para analisar as vendas em comparação com o método manual. Esse ganho de tempo é particularmente vantajoso para produtos de alta rotatividade, nos quais a precisão da ferramenta contribuiu para a redução de falhas de estoque, permitindo uma reposição adequada antes do esgotamento. No entanto, para itens de baixa demanda, a redução de falhas de estoque foi limitada devido à imprevisibilidade nas vendas desses produtos. A ferramenta, portanto, possui potencial para se adequar bem em mercados com menos variedade de itens e vendas mais regulares, sendo uma solução promissora para setores diferentes do de papelaria, onde a previsibilidade de vendas seja maior.

5. CONCLUSÃO

O trabalho explorou as ferramentas e tecnologias relevantes para automação e gerenciamento de estoque, abordando metodologias de análise de dados e algoritmos de previsão. Para isso, utilizamos a linguagem de programação python, que, com bibliotecas como pandas e numpy, permitiu a manipulação e análise eficiente dos dados de transações. Além disso, a interface do streamlit possibilitou uma visualização interativa e acessível dos dados processados, facilitando a compreensão dos padrões de compra e demanda. Embora a análise RFM tenha sido essencial para identificar o valor de cada cliente com base em recência, frequência e valor monetário das compras, optou-se por uma abordagem sem algoritmos de machine learning ou inteligência artificial, focando em técnicas estatísticas e preditivas simplificadas.

Contudo, ainda não foi alcançada uma classificação de prioridade de reposição completamente eficaz, a ferramenta leva muito em conta a quantidade vendida do item, e não tanto a sua real necessidade de reposição, o que limita a aplicação do modelo para um gerenciamento automatizado de estoque. Embora as previsões e insights sobre o comportamento de recompra dos clientes ofereçam uma base sólida para o planejamento de vendas, o sistema de classificação de reposição continua a ser aprimorado para melhor atender às variações de demanda. Essa limitação ressalta a importância de futuros desenvolvimentos que incorporem metodologias avançadas para aprimorar a precisão da previsão de demanda e implementar uma classificação dinâmica e automatizada de reposição de estoque.

Atualmente, o processo de gerenciamento de estoque no e-commerce é realizado de forma manual, utilizando relatórios de vendas extraídos do sistema. Esses relatórios fornecem informações básicas, como os itens vendidos, as quantidades comercializadas e o saldo restante no estoque, que são usados como base para compor uma lista de produtos a serem repostos. Essa lista é então encaminhada aos fornecedores para o reabastecimento. No entanto, essa abordagem manual limita a eficiência e a precisão da reposição, especialmente diante de picos sazonais e variações na demanda, o que torna essencial a implementação de métodos mais automatizados para facilitar o planejamento e a execução das reposições.

A sazonalidade desempenha um papel relevante nas vendas, pois embora a maioria dos produtos possua demanda estável ao longo do ano, determinados períodos — como novembro (Black Friday), dezembro (Natal), e janeiro e fevereiro (volta às aulas) — registram aumentos significativos nas vendas, especialmente de materiais escolares como mochilas e cadernos. Além desses itens sazonais, há produtos de consumo contínuo, como marcadores de quadro

branco, que mantêm um fluxo constante de vendas durante o ano, sendo necessário realizar manutenção frequente do seu estoque. A velocidade de reposição depende tanto da rotatividade de vendas quanto da capacidade dos fornecedores em atender os pedidos rapidamente. Para aprimorar o processo, é fundamental identificar esses fatores críticos e os desafios específicos enfrentados, visando especificar requisitos de sistema que melhorem a agilidade e precisão no gerenciamento do estoque.

O protótipo desenvolvido foi projetado para atender às principais necessidades de análise de estoque e vendas, utilizando como base a análise RFM e séries temporais para prever as vendas futuras em termos de quantidade. Com esses métodos, o sistema analisa os dados dos relatórios e projeta demandas, possibilitando uma gestão de estoque mais fundamentada. No entanto, o sistema de priorização de reposição de itens ainda não atingiu o nível de precisão desejado, e melhorias serão necessárias para que a reposição se adeque mais diretamente às variações de demanda e à sazonalidade.

A aplicação foi desenvolvida em Python, uma escolha fundamentada na robustez da linguagem e nas bibliotecas especializadas em manipulação de dados, como pandas e numpy, que facilitam o processamento e a análise de grandes volumes de dados de vendas. O framework streamlit foi empregado para criar uma interface amigável e acessível, permitindo a visualização interativa dos dados. Embora a integração com plataformas de e-commerce tenha sido planejada, ela não foi implementada neste estágio devido a dificuldades no desenvolvimento. Priorizou-se, assim, o aprimoramento e a otimização do modelo de previsão, visando uma base sólida para melhorias futuras e possível expansão da funcionalidade.

Para validar a precisão do protótipo desenvolvido, foram realizados testes práticos aplicando a ferramenta em situações reais de gestão de estoque e comparando os resultados sugeridos com os dados de vendas reais. Essa análise de eficácia permitiu observar a eficácia do modelo em prever as quantidades de reposição e analisar a eficiência do sistema de priorização implementado. Através desses testes, identificou-se pontos em que o sistema conseguiu captar tendências de demanda, bem como áreas em que a previsão e priorização ainda podem ser otimizadas para um melhor alinhamento com as necessidades de estoque.

5.1 RECOMENDAÇÕES DE TRABALHOS FUTUROS

Integração com Sistemas de CRM e ERP: Explorar a integração do sistema de gerenciamento de estoque com plataformas de CRM (Customer Relationship Management) e ERP (Enterprise Resource Planning) para criar um fluxo de dados entre estoque, vendas e atendimento ao cliente, proporcionando uma visão integrada e otimizada da operação.

Desenvolvimento de Módulos de Inteligência Artificial Avançada: Implementar algoritmos de machine learning mais avançados, como redes neurais, para previsões de demanda ainda mais precisas, identificando padrões complexos de vendas e antecipando variações sazonais.

Expansão para Análise de Dados em Tempo Real: Investigar a possibilidade de aplicar tecnologias de análise em tempo real, permitindo que o sistema reaja imediatamente às flutuações do mercado e ajuste as recomendações de reposição de forma instantânea.

Automatização da Reposição de Estoque com Fornecedores: Desenvolver uma extensão que possibilite ao sistema acionar automaticamente pedidos de reposição com fornecedores, baseado no nível de estoque e previsão de demanda, facilitando a reposição contínua.

Incorporação de Funcionalidades de Análise de Concorrência: Ampliar o sistema para analisar dados de concorrência, como preços e disponibilidade, para ajustar estratégias de reposição e preços com base na competitividade de mercado.

Aplicação da Solução em Diversos Nichos de Mercado: Realizar estudos sobre a aplicabilidade e adaptações do sistema para diferentes setores do e-commerce, como moda, eletrônicos e alimentos, que possuem demandas e desafios específicos na gestão de estoque.

Otimização do visual da saída de dados: Formatação adequada do conjunto de dados gerados para um formato PDF com uma visibilidade mais limpa e clara, permitindo um entendimento pleno dos dados por qualquer usuário e garantindo a plena usabilidade do sistema em um ambiente não especializado.

REFERÊNCIAS

ALVES, William P. **Linguagem e Lógica de Programação**. São Paulo: SRV Editora LTDA, 2013. *E-book.*

_____. **Programação Python: aprenda de forma rápida**. Rio de Janeiro: Expressa, 2021. *E-book.*

AWARI. Streamlit: **A Biblioteca Python que Revoluciona o Desenvolvimento de Aplicações Web.** Disponível em: https://awari.com.br/streamlit-python/?utm_source=blog. Acessado em 16 de nov. de 2024.

BEHRMAN, Kennedy R. **Fundamentos de Python para ciência de dados**. Porto Alegre: Bookman, 2023. *E-book.*

DOCKER, **Documento de software**. Disponível em: < https://www.docker.com/>. Acesso em: 30 jun. 2024.

DUCKETT, Jon. **PHP&MYSQL: desenvolvimento web no lado do servidor**. Rio de Janeiro: Editora Alta Books, 2024. *E-book.*

FREITAS, Pedro Henrique C.; BIRNFELD, Karine; SARAIVA, Maurício de O.; et al. **Programação Back End III**. Porto Alegre: Grupo A, 2021. *E-book.*

GITHUB, **Documento de software**. Disponível em: < https://docs.github.com/pt>. Acesso em: 30 jun. 2024.

Medium. Lifetimes: **Customer Lifetime Value with Python´s Lifetimes Library.** Disponível em: https://medium.com/@mine.gazioglu40/customer-lifetime-value-with-pythons-lifetimes-library-d964158b35b3. Acessado em 16 de nov. de 2024.

MILETTO, Evandro M.; BERTAGNOLLI, Silvia C. **Desenvolvimento de software II: introdução ao desenvolvimento web com HTML, CSS, javascript e PHP. (Tekne)**. Porto Alegre: Grupo A, 2014. *E-book.*

MORAIS, Izabelly S.; ZANIN, Aline. **Engenharia de software**. Porto Alegre: Grupo A, 2020. *E-book.*

PAULA FILHO, Wilson de Pádua. **Engenharia de Software - Projetos e Processos - Vol. 2**. São Paulo: Grupo GEN, 2019. *E-book.*

PYTHON, **Documento de software**. Disponível em: < https://www.python.org//>. Acesso em: 16 nov. 2024.

SANTOS, Marcela G.; SARAIVA, Maurício O.; FÁTIMA, Priscila G. **Linguagem de programação**. Porto Alegre: Grupo A, 2018. *E-book.*

SBROCCO, José Henrique Teixeira de C.; MACEDO, Paulo Cesar de. **Metodologias Ágeis - Engenharia de Software sob Medida**. São Paulo: SRV Editora LTDA, 2012. *E-book.*

SILVA, Fernanda R.; SOARES, Juliane A.; SERPA, Matheus da S.; et al. **Cloud Computing**. Porto Alegre: Grupo A, 2020. *E-book.*

VETORAZZO, Adriana S. **Engenharia de software**. Porto Alegre: Grupo A, 2018. *E-book.*