

# Sistema de análise de sentimentos para avaliações de produtos de e-commerce: uma ferramenta de apoio à decisão do consumidor\*

Sentiment analysis system for e-commerce product reviews: a consumer decision support tool

Wesley Henrique de Lima<sup>1</sup>  
Gustavo Luís Soares (Orientador)<sup>2</sup>

## Resumo

O crescimento exponencial do comércio eletrônico gerou um volume massivo de avaliações de consumidores, criando o desafio de processar essas informações para uma tomada de decisão eficiente. Este trabalho apresenta o desenvolvimento de um sistema computacional para coleta, análise de sentimentos e visualização de *insights* de produtos de informática, com o objetivo de auxiliar o consumidor na tomada de decisão de compra. A metodologia adotou uma abordagem aplicada e incremental, utilizando a técnica de *web scraping* com Selenium para a extração de dados da plataforma Kabum, seguida pelo *fine-tuning* supervisionado de um modelo de linguagem pré-treinado, fundamentado na arquitetura RoBERTa, para a classificação de polaridade das avaliações. O sistema integrou um *backend* desenvolvido com FastAPI e banco de dados MongoDB a uma interface web interativa em Next.js. Os resultados obtidos validam a eficácia da proposta, com a coleta bem-sucedida de mais de 53 mil avaliações e o modelo alcançando uma acurácia de 91,50% e um *F1-score* de 90,77% no conjunto de validação, apesar das limitações impostas pelo desbalanceamento das classes. O sistema atua como uma ferramenta robusta de apoio à decisão, demonstrando a aplicabilidade prática de técnicas de Recuperação de Informação e Inteligência Artificial no contexto do varejo digital.

**Palavras-chave:** Análise de Sentimentos. *Web Scraping*. *E-commerce*. Processamento de Linguagem Natural. Apoio à Decisão.

---

\*Trabalho de conclusão de curso, Sistemas de Informação, Unidade São Gabriel

<sup>1</sup>Graduando de Sistemas de Informação pela PUC Minas, Brasil– wesley.lima@sga.pucminas.br

<sup>2</sup>Instituto de Ciências Exatas e de Informática da PUC Minas, Brasil– gsoares@pucminas.br

### Abstract

The exponential growth of e-commerce has generated a massive volume of consumer reviews, creating a challenge in processing this information for efficient decision-making. This work presents the development of a computational system for the collection, sentiment analysis, and visualization of insights regarding computer products, aiming to assist consumers in their purchasing decisions. The methodology adopted an applied and incremental approach, utilizing the web scraping technique with Selenium to extract data from the Kabum platform, followed by the supervised fine-tuning of a pre-trained language model based on the RoBERTa architecture for the polarity classification of reviews. The system integrates a backend developed with FastAPI and a MongoDB database into an interactive web interface built with Next.js. The results obtained validate the effectiveness of the proposal, with the successful collection of over 53,000 reviews and the model achieving an accuracy of 91.50% and an F1-score of 90.77% on the validation set, despite the limitations imposed by class imbalance. The system acts as a robust decision support tool, demonstrating the practical applicability of Information Retrieval and Artificial Intelligence techniques in the context of digital retail.

**Keywords:** Sentiment Analysis. Web Scraping. E-commerce. Natural Language Processing. Decision Support.

## 1 INTRODUÇÃO

O crescimento significativo do *e-commerce* transformou profundamente os hábitos de consumo da população e estabeleceu sua marca na economia digital moderna. De acordo com dados do Sebrae em 2023, mais de 61% dos consumidores brasileiros já demonstram preferência por compras online, mostrando a importância do *e-commerce* nas decisões de compra (SEBRAE, 2023). Atualmente, é comum que clientes, antes de finalizarem uma compra, busquem por avaliações e opiniões de outros compradores para apoiar sua decisão. Esse cenário gerou um volume massivo de dados não estruturados, principalmente na forma de avaliações e comentários de produtos deixados pelos usuários (FILHO; LIMONGI; MARTINS, 2025). Embora essas opiniões constituam fontes importantes de informação tanto para consumidores quanto para empresas, analisar manualmente milhares de avaliações para extrair resultados úteis é uma tarefa impraticável e complexa para o consumidor médio, tornando o processo de escolha cansativo e ineficaz.

Nesse contexto, a Análise de Sentimentos, também conhecida como Mineração de Opinião, surge como uma ferramenta estratégica essencial. Ela permite a extração e classificação automática de opiniões de grandes quantidades de texto, transformando dados brutos em pontos de vista, avaliações, emoções e atitudes relacionadas às entidades, gerando, assim, *insights* essenciais (FILHO; LIMONGI; MARTINS, 2025). A relevância desta abordagem é particularmente alta para os consumidores de *e-commerce* em produtos de informática, um público que frequentemente enfrenta um volume massivo de avaliações técnicas e contraditórias, tornando a escolha de um produto uma tarefa complexa e demorada. Para este consumidor, a implementação de um sistema capaz de coletar, processar e apresentar de forma intuitiva a análise dessas avaliações oferece uma poderosa ferramenta de apoio à decisão. Conforme apontado por Rodrigues, Flávio e Neto (2021), sistemas dessa natureza auxiliam a economizar tempo e a aumentar a confiança e a segurança de compra, resolvendo diretamente o desafio da sobrecarga informacional.

O objetivo geral deste trabalho é desenvolver e validar um sistema computacional para coleta, análise de sentimentos e visualização de dados de produtos de *e-commerce*, avaliando a eficácia de modelos de linguagem ajustados (*fine-tuned*) para o nicho de informática, com a finalidade de auxiliar os consumidores na tomada de decisão a respeito de novas compras. Considerando este propósito, foram definidos os seguintes objetivos específicos: (1) implementar um processo de coleta de dados via *web scraping* para extrair informações e avaliações de produtos de informática de uma plataforma de *e-commerce*; (2) realizar o ajuste fino (*fine-tuning*) e validação de um modelo de linguagem pré-treinado (RoBERTa) para a classificação de sentimentos; (3) implementar uma API REST destinada à exposição e distribuição dos dados processados, assegurando a interoperabilidade e o desacoplamento entre o núcleo de análise e a interface de visualização; e (4) construir uma interface de visualização que sintetize métricas quantitativas e qualitativas, transformando dados brutos em indicadores visuais de apoio à decisão.

Este artigo está estruturado da seguinte forma: esta introdução contextualiza o problema, a justificativa e os objetivos da pesquisa. A Seção 2, Referencial Teórico, aborda os conceitos fundamentais de análise de sentimentos, Processamento de Linguagem Natural, a arquitetura de modelos Transformer, tecnologias de *web scraping* e trabalhos relacionados. A Seção 3 descreve a Metodologia e a abordagem incremental adotada. A Seção 4 detalha o Desenvolvimento técnico e a implementação da solução. A Seção 5 apresenta e discute os Resultados obtidos. Por fim, a Seção 6 traz a Conclusão do estudo, seguida pelas Referências Bibliográficas.

## 2 REFERENCIAL TEÓRICO

Nesta seção, são abordadas as principais tecnologias e conceitos utilizados para o desenvolvimento do sistema proposto.

### 2.1 Coleta de Dados com *Web Scraping*

O *web scraping* ou raspagem de dados é uma técnica computacional na qual um programa realiza a automatização da extração de dados dos websites, processo realizado através da recuperação e análise de seu conteúdo HTML. Trata-se de uma técnica fundamental quando não há uma API (Interface de Programação de Aplicações) disponível para seu consumo e integração (SOUZA *et al.*, 2021).

O procedimento consiste em requisitar a informação de um servidor, analisar a resposta e armazenar os dados de forma estruturada para uso posterior. Muitos sites modernos carregam conteúdo dinamicamente com JavaScript, o que torna a extração de dados mais complexa. Em casos como este, utilizam-se ferramentas como o Selenium, que automatizam um navegador para renderizar a página completamente, incluindo o conteúdo gerado por *scripts* (MITCHELL, 2025). Essa técnica é aplicada de forma prática na coleta de informações e avaliações de produtos em plataformas de *e-commerce*.

A prática de *web scraping* exige uma postura ética e responsável. É uma boa prática respeitar as diretrizes do arquivo *robots.txt*, uma convenção utilizada por administradores de sites para indicar quais seções não devem ser acessadas por robôs. Além disso, para evitar a sobrecarga dos servidores e possíveis bloqueios, é recomendado simular as técnicas de comportamento humano, como ajustes nos cabeçalhos das requisições HTTP e pausas entre as requisições (MITCHELL, 2025).

## 2.2 Processamento de Linguagem Natural (PLN)

O Processamento de Linguagem Natural (PLN) é uma área da inteligência artificial cujo objetivo principal é permitir que máquinas processem e compreendam a linguagem humana. Visto que a comunicação humana é inerentemente complexa, ambígua e cheia de nuances, é necessário realizar um pré-processamento nos textos coletados antes de submetê-los a uma análise (RODRIGUES; FLÁVIO; NETO, 2021). Essa etapa de preparação, que inclui a limpeza e estruturação dos dados, é crucial para a eficácia dos modelos.

O pré-processamento de texto inclui diversas etapas fundamentais para a preparação dos dados. As mais comuns são a conversão para minúsculas, que unifica o texto para que palavras com diferentes capitalizações sejam tratadas como o mesmo termo. A tokenização segmenta o texto em unidades menores (*tokens*), que são as unidades básicas de análise. Em seguida, a remoção de *stopwords* elimina palavras comuns que possuem baixo valor semântico. Por fim, a lematização reduz as palavras à sua forma raiz ou lema, considerando o contexto morfológico. Por exemplo, “correndo” é reduzido para “correr” (FILHO; LIMONGI; MARTINS, 2025).

A execução correta dessas etapas é fundamental para aumentar a acurácia dos modelos de classificação, garantindo que a análise se concentre nos termos que de fato expressam sentimento (RODRIGUES; FLÁVIO; NETO, 2021).

## 2.3 Análise de Sentimentos

A análise de sentimentos, também conhecida como mineração de opinião, emprega o Processamento de Linguagem Natural (PLN), análise textual e linguística computacional para reconhecer, extrair e categorizar estados emocionais e dados subjetivos. A finalidade é identificar o sentimento predominante de um texto, classificando-o como positivo, negativo ou neutro. (SOUZA *et al.*, 2021).

Com o avanço das mídias sociais e do *e-commerce*, as opiniões dos consumidores se tornaram um bem muito valioso para as empresas, que buscam automatizar o processo de monitoramento de suas marcas e produtos (SOUZA *et al.*, 2021). Para os consumidores, a análise de sentimentos serve como uma ferramenta poderosa de apoio à decisão, permitindo avaliar as vantagens e desvantagens de um produto com base no *feedback* de outros compradores, o que os auxilia a realizar escolhas mais seguras e informadas (RODRIGUES; FLÁVIO; NETO, 2021).

## 2.4 Modelos Transformers

A arquitetura Transformer demonstrou uma mudança fundamental no Processamento de Linguagem Natural, servindo como base para os modernos modelos de linguagem, como o BERT (Bidirectional Encoder Representations from Transformers). Diferente das arquiteturas

ras recorrentes que processam texto de forma sequencial, o Transformer é capaz de receber e analisar o texto de entrada de uma só vez, utilizando mecanismos de atenção para ponderar a importância de diferentes palavras na sequência (SILVA *et al.*, 2024).

O BERT é um modelo de linguagem que se destaca por sua capacidade de processar o texto de forma bidirecional, ou seja, analisando cada palavra em relação a todas as outras na frase, tanto à esquerda quanto à direita. Essa abordagem permite uma compreensão contextual mais profunda e apurada do significado das palavras, superando as limitações dos modelos unidirecionais anteriores (FILHO; LIMONGI; MARTINS, 2025).

A aplicação de um modelo como o BERT consiste em duas etapas principais: o pré-treino, no qual o modelo aprende representações da linguagem a partir de um vasto volume de texto não rotulado, e o ajuste fino (*fine-tuning*), onde o modelo é adaptado com seus parâmetros pré-treinados para uma tarefa específica, como a análise de sentimentos, utilizando dados rotulados (SILVA *et al.*, 2024). Uma evolução significativa desta arquitetura é o RoBERTa (Robustly Optimized BERT Pretraining Approach), que otimiza os hiperparâmetros de treinamento do BERT para obter desempenho superior. Além da arquitetura, a adaptação ao domínio é crucial. Nesse sentido, Loureiro *et al.* (2022) demonstram que modelos derivados do RoBERTa, quando treinados continuamente com dados de redes sociais (como o Twitter), desenvolvem uma capacidade superior de lidar com a dinamicidade da linguagem informal e com novos contextos temporais.

## 2.5 Interface de Programação de Aplicações

A Interface de Programação de Aplicações (API) é um conjunto de definições e protocolos que permite a comunicação e a integração entre diferentes softwares. No contexto desta aplicação, ela atua como um componente essencial do *backend*, funcionando como o núcleo responsável por implementar a lógica de negócio que gerencia o fluxo de dados do sistema. Em uma arquitetura de software moderna, a API atua como uma interface que permite a comunicação entre diferentes componentes de software. O padrão arquitetural REST (Representational State Transfer) fundamenta-se em uma estrutura em camadas e na característica *stateless*, onde cada requisição é tratada de forma isolada, sem que o servidor armazene o estado do cliente. Nesse contexto, APIs REST atuam frequentemente como intermediárias entre a camada de persistência de dados e as interfaces de consumo, padronizando a troca de informações, comumente através do formato JSON (ALEKSENDRIĆ *et al.*, 2024).

A API é frequentemente implementada com o FastAPI, um *framework* web moderno para Python, projetado para a construção de APIs de alto desempenho. Sua performance é derivada da base que permite o processamento de requisições de forma assíncrona. A arquitetura do FastAPI utiliza intensivamente as anotações de tipo (*type hints*) do Python em conjunto com a biblioteca Pydantic para realizar a validação automática e robusta de dados. Possui uma compatibilidade com padrões do OpenAPI que viabiliza a geração automática de documentação

interativa da API, o que simplifica os processos de teste e integração (ALEKSENDRIĆ *et al.*, 2024).

## 2.6 Interface Web de Visualização

A interface web de visualização compõe o *frontend* da aplicação, sendo a camada responsável pela interação direta com o usuário. Para o desenvolvimento de interfaces modernas, interativas e eficientes, bibliotecas como o React.js se destacam como uma das tecnologias mais populares. O React permite a construção de aplicações de página única (SPAs) com uma experiência de uso fluida, semelhante à de um aplicativo desktop. Sua principal abordagem é a arquitetura orientada a componentes, que consiste em dividir a interface em partes menores, reutilizáveis e independentes. Essa modularidade não apenas simplifica o desenvolvimento e a manutenção de Interfaces do Usuário (UIs) complexas, mas também melhora a performance por meio de um “DOM virtual”, que otimiza as atualizações na tela e garante uma resposta rápida às interações do usuário (ALEKSENDRIĆ *et al.*, 2024).

A criação dos componentes React é predominantemente realizada através do JSX, uma extensão de sintaxe que combina JavaScript com uma marcação similar ao HTML. Embora não seja estritamente obrigatório, o uso do JSX é o padrão no desenvolvimento com React, pois torna a declaração das visões mais intuitiva. Para gerenciar a interatividade e os dados que mudam ao longo do tempo, como filtros de busca ou os resultados de uma análise, o React utiliza “*Hooks*”. Os *Hooks* são funções que permitem aos desenvolvedores acessar o ciclo de vida de um componente para manipular seu estado, tudo isso sem a necessidade de utilizar classes. Essas abordagens simplificam o gerenciamento de componentes complexos e viabilizam a reutilização de lógica entre eles, como na comunicação com uma API para buscar os dados da análise de sentimentos e exibi-los ao consumidor (ALEKSENDRIĆ *et al.*, 2024).

## 2.7 Trabalhos Relacionados

Rodrigues, Flávio e Neto (2021) propõem um sistema que utiliza a análise de sentimentos como uma estratégia comercial para avaliar o nível de satisfação dos clientes de um *e-commerce*. Assim como neste trabalho, os autores utilizam a técnica de *web scraping* para a coleta de dados de produtos e seus respectivos comentários. A análise de sentimentos é aplicada para extrair a polaridade das opiniões (positiva, negativa ou neutra), transformando o *feedback* dos consumidores em um indicador de satisfação. Este estudo é de grande relevância por abordar o mesmo domínio de *e-commerce* e compartilhar o objetivo de transformar opiniões de consumidores em *insights* valiosos, embora o foco principal seja a estratégia comercial para as empresas, enquanto o presente trabalho se concentra em ser uma ferramenta de apoio direto ao consumidor.

Souza *et al.* (2021) exploram a aplicação da análise de sentimentos em um domínio diferente, o monitoramento da imagem de instituições de ensino superior nas redes sociais. O trabalho também emprega técnicas de *web scraping* para a coleta de comentários e utiliza serviços de análise de sentimentos para gerar indicadores de satisfação que podem apoiar estratégias de marketing. Embora o contexto seja distinto (educacional *versus* comercial), as técnicas de coleta e análise de dados são análogas às utilizadas no sistema proposto, demonstrando a versatilidade e a aplicabilidade da análise de sentimentos em diferentes setores.

Por fim, Silva *et al.* (2025) apresentam o desenvolvimento de uma plataforma de recomendação inteligente para *e-commerce* que integra modelos de linguagem natural (LLMs ou Modelos de Linguagem de Grande Escala) e serviços em nuvem. A coleta de dados também é realizada por meio de *web scraping* em *marketplaces* de informática. Embora o objetivo final seja um sistema de recomendação de produtos e não uma ferramenta focada na análise de avaliações existentes, o trabalho é relevante por atuar no mesmo nicho de mercado (produtos de informática) e utilizar uma abordagem tecnológica similar para a extração de dados e análise de sentimentos. Isso evidencia a tendência de aplicação de técnicas de PLN e coleta automatizada de dados para aprimorar a experiência do usuário no varejo digital.

Os trabalhos validam a relevância e a viabilidade das técnicas de *web scraping* e análise de sentimentos no contexto do *e-commerce*. Este projeto se diferencia por focar especificamente na criação de uma ferramenta de apoio à decisão para o consumidor, consolidando e visualizando de forma clara e intuitiva o sentimento geral extraído das avaliações de produtos, proporcionando ao comprador um poder de decisão com base em *insights* claros.

### 3 METODOLOGIA

Nesta seção, são apresentados os procedimentos e métodos utilizados para a realização do desenvolvimento do sistema proposto. Trata-se de uma pesquisa aplicada, pois utiliza conceitos como *web scraping*, análise de sentimentos e desenvolvimento de software para gerar conhecimentos práticos, com a finalidade de solucionar problemas relacionados ao grande volume de dados em avaliações de produtos e auxiliar a decisão de compra do consumidor. O desenvolvimento do sistema, denominado *Review Sentimentum*, segue uma abordagem incremental, dividida em quatro fases principais: (1) Coleta e Armazenamento de Dados; (2) Análise de Sentimentos; (3) Desenvolvimento da API; e (4) Desenvolvimento da Interface de Visualização do Usuário.

Para a execução de todas as etapas computacionais, englobando a coleta automatizada de dados, o processamento do modelo de linguagem e a execução da aplicação em ambiente local, utilizou-se uma estação de trabalho equipada com processador Intel Core i5-12400F (6 núcleos, 12 *threads*, até 4,40 GHz) e 16 GB de memória RAM DDR4. O processamento do modelo de linguagem foi acelerado por uma GPU NVIDIA GeForce RTX 4060 (8 GB GDDR6, 3072 núcleos CUDA), e o armazenamento dos dados foi realizado em um SSD NVMe M.2 de 500



GB (3500 MB/s de leitura), garantindo a performance necessária para a manipulação do volume de dados.

### 3.1 Coleta e Armazenamento de Dados

A primeira etapa de coleta e armazenamento de dados do projeto consistiu na obtenção dos dados brutos de avaliações de produtos. Foi selecionada como fonte de dados a plataforma de *e-commerce* brasileira Kabum, devido ao seu volume significativo de avaliações detalhadas em produtos de informática. Esta fase é crucial, pois a qualidade e a quantidade dos dados coletados impactam diretamente a precisão dos modelos de análise de sentimentos a serem aplicados. A escolha da fonte visa, portanto, garantir um conjunto de informações robusto e diretamente alinhado ao objetivo de auxiliar consumidores deste nicho de mercado.

Para a extração automatizada, optou-se pela técnica de *web scraping*. A ferramenta principal utilizada foi a biblioteca Selenium em conjunto com a linguagem Python. A escolha do Selenium justifica-se pela capacidade de interagir com elementos dinâmicos da página (renderizados via JavaScript), que não seriam capturados por requisições HTTP estáticas. O processo de coleta foi desenvolvido para extrair as informações dos produtos e o conjunto de avaliações dos usuários.

### 3.2 Análise de Sentimentos

Para a classificação das avaliações, adotou-se a estratégia de *Transfer Learning*, utilizando como base o modelo pré-treinado Caramelo Smile (Adilmar Coelho Dantas, 2024). Trata-se de um modelo de linguagem com mais de 100 milhões de parâmetros, derivado da arquitetura RoBERTa (LOUREIRO *et al.*, 2022), que foi originalmente ajustado para a tarefa de análise de sentimentos em português. A escolha deste modelo como ponto de partida justifica-se pela sua especialização linguística e arquitetura robusta. Por ser nativo para o português do Brasil, ele demonstra capacidade superior de compreender estruturas gramaticais locais em comparação a modelos multilíngues genéricos. Além disso, sua performance base, com acurácia validada de 90,52% em *benchmarks* gerais, oferece um ponto de partida confiável para a especialização.

Entretanto, dado que a linguagem utilizada em avaliações de produtos de informática possui particularidades incluindo termos técnicos e gírias de hardware, o modelo base foi submetido a uma nova etapa de *fine-tuning* supervisionado realizada especificamente para este trabalho. Nesta fase, o modelo foi re-treinado com o *dataset* de avaliações da Kabum coletado e rotulado durante o projeto, permitindo que a rede neural ajustasse seus pesos para compreender o contexto específico do nicho de tecnologia e aumentar a precisão da classificação. A implementação técnica desse treinamento e da aplicação do modelo foi realizada por meio da

biblioteca Transformers. Para mensurar a eficácia deste ajuste fino e validar a capacidade de generalização do modelo, foram estabelecidas métricas quantitativas padronizadas. A primeira métrica adotada foi a acurácia (*accuracy*), que representa a taxa de acerto global do sistema, ou seja, de todas as avaliações analisadas, quantas o modelo classificou corretamente.

No entanto, considerando o desbalanceamento natural das avaliações de *e-commerce*, onde a classe “positiva” é majoritária, olhar apenas para a taxa de acerto global poderia mascarar deficiências do modelo em identificar críticas negativas. Para mitigar esse viés e obter uma avaliação mais justa, utilizou-se o *F1-score*. Esta métrica funciona como um equilíbrio entre dois outros indicadores fundamentais: a precisão e a revocação. Em termos práticos, a precisão indica a confiabilidade do modelo: quando ele classifica uma avaliação como “positiva”, qual a chance de ele estar realmente certo (evitando falsos alarmes). Já a revocação (ou sensibilidade) mede a capacidade de varredura do sistema: de todas as avaliações positivas que existem, quantas o modelo conseguiu encontrar sem deixar passar despercebidas. O *F1-score*, portanto, resume esses dois aspectos, garantindo que o sistema seja, ao mesmo tempo, preciso e abrangente. Adicionalmente, o processo de aprendizado foi monitorado através da função de perda *Cross-Entropy Loss*. Diferente das métricas anteriores que avaliam o acerto final, esta métrica atua como um “termômetro de erro” durante o treinamento: ela calcula o quão distante e confusa estava a previsão do modelo em relação à resposta correta, buscando minimizar essa incerteza a cada ciclo de aprendizado.

### 3.3 Desenvolvimento da API

Para garantir uma arquitetura desacoplada e modular, onde o processamento de dados é independente da interface de usuário, foi desenvolvida uma API seguindo o padrão REST. A implementação foi realizada em Python com o *framework* FastAPI, uma escolha estratégica motivada por sua alta performance e sustentada pela facilidade na criação de *endpoints* robustos e bem documentados. A principal responsabilidade da API é servir como uma ponte entre o banco de dados e as aplicações clientes, consultando a base de dados NoSQL MongoDB para recuperar as informações processadas, como os dados dos produtos e as classificações de sentimento. Assim disponibilizá-las de forma padronizada no formato JSON, que é leve e facilmente consumível por interfaces web.

Para atender aos requisitos funcionais do sistema, a interface foi projetada para cobrir a jornada completa de descoberta de produtos. Foram definidos *endpoints* que permitem a navegação estruturada pela hierarquia de categorias do *e-commerce*, a busca textual com paginação para refino de resultados e a recuperação detalhada de um item específico pelo seu identificador. Essa estrutura visa garantir que todos os dados processados e os *insights* de sentimento sejam entregues de forma eficiente para a interface do usuário.

### 3.4 Desenvolvimento da Interface de Visualização do Usuário

A etapa final do desenvolvimento consistiu na criação de uma interface web interativa, projetada para ser o ponto de contato direto e interação com o consumidor final. A aplicação foi desenvolvida com o *framework* Next.js, que se baseia na biblioteca React.js, utilizando a linguagem TypeScript, que adiciona um sistema de tipagem estática ao código, conferindo maior robustez e facilitando a detecção de erros durante o desenvolvimento. Essa tecnologia foi selecionada por sua notável eficiência na construção de interfaces de usuário reativas e componentizadas. Essa abordagem permitiu o desenvolvimento de uma interface modular e de fácil manutenção, onde elementos web como cartões de produtos podem ser reutilizados.

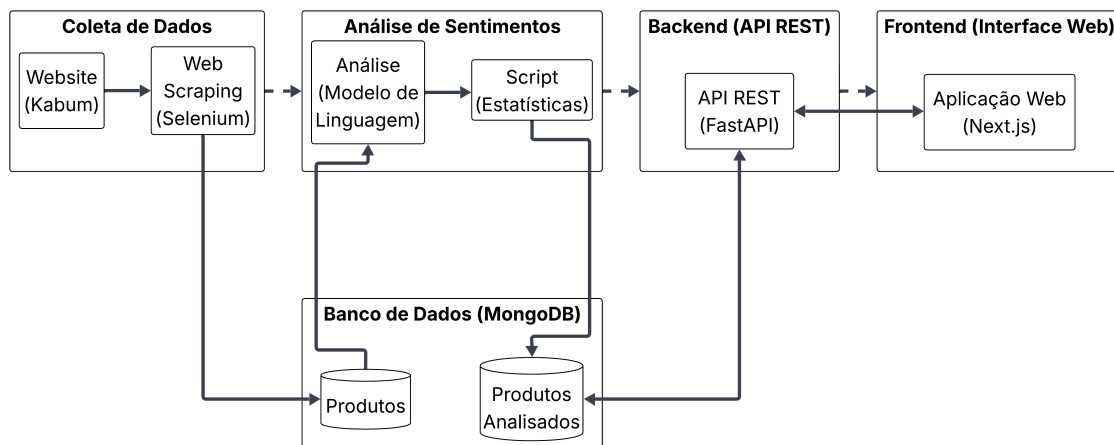
A interface foi projetada para apresentar os resultados da análise de forma clara e intuitiva. Para isso, foram implementados os seguintes requisitos funcionais: uma página inicial multifuncional, que oferece mecanismos de busca flexíveis (por texto, código ou *link*) e um menu de navegação por categorias; páginas de listagem que exibem os resultados ou categorias selecionadas através de *cards* informativos; uma página institucional dedicada a apresentar o *Review Sentimentum* e fornecer contextualização ao usuário; e uma página de produto detalhada, que exibe os dados consolidados da análise de sentimentos por meio de elementos visuais.

## 4 DESENVOLVIMENTO

O desenvolvimento do sistema foi realizado a partir de um formato de *pipeline*, organizado em quatro etapas sequenciais. A base tecnológica do projeto foi a *stack* FARM (FastAPI, React e MongoDB) de acordo com (ALEKSENDRIĆ *et al.*, 2024), selecionada por sua modernidade e por promover uma arquitetura desacoplada, onde as responsabilidades entre a interface do cliente, a lógica de negócio e o armazenamento de dados são claramente divididas, garantindo modularidade e facilidade de manutenção.

A implementação seguiu a sequência, iniciando com a coleta automatizada de avaliações, seguida pelo processamento e análise de sentimentos dos textos. Os resultados foram então disponibilizados por meio de uma API REST, que serve como ponte para a interface web interativa. Essa abordagem estruturada permitiu o desenvolvimento focado de cada componente do sistema, cujos detalhes técnicos e decisões de implementação serão aprofundados nos tópicos subsequentes desta seção. A arquitetura geral do sistema proposto é ilustrada na Figura 1. Para fins de consulta, transparência e reprodutibilidade, todo o código-fonte do projeto, bem como as instruções para sua execução, foram disponibilizados publicamente em repositório no GitHub (LIMA, 2025).

**Figura 1 – Diagrama da arquitetura do sistema de análise de sentimentos.**



Fonte: Elaborado pelo autor.

#### 4.1 Implementação da Coleta e Armazenamento de Dados

A etapa de extração de dados foi feita por meio de *web scraping*, utilizando a biblioteca Selenium (Selenium Project, 2025) com Python para automatizar a navegação e interagir com o conteúdo dinâmico do site da Kabum. O processo iniciou-se com uma análise manual do site, incluindo a verificação do arquivo *robots.txt* para garantir a conformidade com as diretrizes de acesso e o mapeamento da estrutura de URLs a partir do *sitemap.xml*. Foram desenvolvidos *scripts* para percorrer as páginas de produtos de hardware, utilizando seletores CSS e XPath para identificar e extrair informações relevantes, como detalhes do produto e avaliações de usuários. Para simular o comportamento humano e evitar sobrecarga no servidor, foram implementadas pausas aleatórias entre as requisições e um sistema de tratamento de erros para garantir a robustez da coleta.

Os dados extraídos foram inseridos em um banco de dados NoSQL MongoDB. A comunicação com o banco foi realizada através do *driver* PyMongo (MongoDB, Inc., 2025), que permitiu a criação de uma estrutura organizada em coleções distintas: uma para armazenar as URLs obtidas do *sitemap*, outra para os dados detalhados dos produtos e suas respectivas avaliações, e uma terceira para registrar estatísticas do processo de extração. Adicionalmente, foi implementado um sistema de logs para registrar o andamento da execução, incluindo sucessos e falhas, assegurando o rastreamento e a integridade do processo de coleta.

#### 4.2 Implementação da Análise de Sentimentos

A etapa de análise de sentimentos foi executada através do ajuste *fine-tuning* do modelo de linguagem pré-treinado Caramelo Smile, visando adaptá-lo ao nicho específico de avaliações

de produtos de hardware. Para a realização do experimento de *fine-tuning*, partiu-se de um total de 53.307 avaliações brutas coletadas. Deste montante, foi extraída uma amostragem aleatória de 1.996 avaliações para a construção do *dataset* de treinamento supervisionado. Estas avaliações foram rotuladas manualmente, resultando em uma distribuição de classes que reflete a realidade do *e-commerce*, conforme detalhado na Tabela 1. O subconjunto anotado foi dividido estrategicamente na proporção de 80% para treino e 20% para validação.

**Tabela 1 – Distribuição das classes nos conjuntos de treinamento e validação**

Subconjunto	Positivas	Neutras	Negativas	Total
Treinamento (80%)	1.429	101	66	1.596
Validação (20%)	358	26	16	400
<b>Total (Amostra)</b>	<b>1.787</b>	<b>127</b>	<b>82</b>	<b>1.996</b>

Fonte: Elaborado pelo autor.

O processo de *fine-tuning* foi implementado utilizando a biblioteca Transformers da Hugging Face (Hugging Face, 2025), configurado com os hiperparâmetros detalhados na Tabela 2. O treinamento foi processado em ambiente local com aceleração por GPU, otimizando o tempo de convergência.

**Tabela 2 – Hiperparâmetros do Treinamento**

Hiperparâmetro	Valor Configurado
Épocas ( <i>Epochs</i> )	3
<i>Batch Size</i> (Treino/Eval)	8
Taxa de Aprendizado ( <i>Learning Rate</i> )	2e-5
<i>Weight Decay</i>	0,01
Estratégia de Avaliação	Por época

Fonte: Elaborado pelo autor.

Após a geração do modelo customizado, foi implementado o script para a execução da análise no sistema. O processo inicia carregando as avaliações restantes a partir da coleção original no MongoDB. Em seguida, utiliza-se um *pipeline* da biblioteca Transformers para classificar os textos em lotes, garantindo a eficiência no uso da GPU durante a inferência. Para cada produto processado, um outro script calcula métricas estatísticas consolidadas, incluindo a contagem e o percentual de cada polaridade (positivo, negativo, neutro), a média de confiança do modelo por categoria e a correlação entre a nota original dada pelo usuário e o sentimento classificado. Por fim, os dados enriquecidos com esses *insights* são armazenados em um novo banco de dados no MongoDB. Essa abordagem de segregação preserva a integridade dos dados brutos coletados inicialmente, enquanto organiza os dados processados para uma consulta performática pela API.

### 4.3 Desenvolvimento da API REST

A construção da API REST foi realizada com o *framework* FastAPI (RAMÍREZ, 2025) em Python, uma escolha estratégica devido à sua alta performance e ao suporte nativo para operações assíncronas. Para garantir a manutenibilidade e a separação de responsabilidades, foi adotada uma arquitetura multicamadas, organizada em cinco diretórios principais: *core*, *models*, *services*, *routers* e *utils*. A camada *core* centralizou as configurações da aplicação e a gestão da conexão assíncrona com o banco de dados MongoDB. A camada *routers* foi responsável por definir os *endpoints* e gerenciar as requisições HTTP, enquanto a camada *services* encapsulou toda a lógica de negócio e as operações de manipulação de dados. Para evitar a duplicação de código, foi introduzido o diretório *utils*, destinado a funções auxiliares e transversais ao sistema. Nesta pasta, foram implementados métodos utilitários, como funções específicas para o tratamento de *strings*.

Para a estruturação e validação dos dados, a camada *models* utilizou a biblioteca Pydantic, que se integra nativamente ao FastAPI para garantir a integridade dos dados trafegados entre o cliente e o servidor. Foram desenvolvidos *endpoints* estratégicos para suportar a interatividade do sistema, incluindo rotas para a navegação progressiva entre os níveis de categorias e a listagem de produtos por localização específica no catálogo. Complementarmente, implementou-se um mecanismo de busca textual paginado e uma rota dedicada à recuperação completa de um produto por seu identificador único (*ID*). Desse modo, a API atua como uma ponte eficiente e desacoplada, disponibilizando os dados processados e os *insights* da análise de sentimentos em formato JSON padronizado para consumo pela interface de visualização.

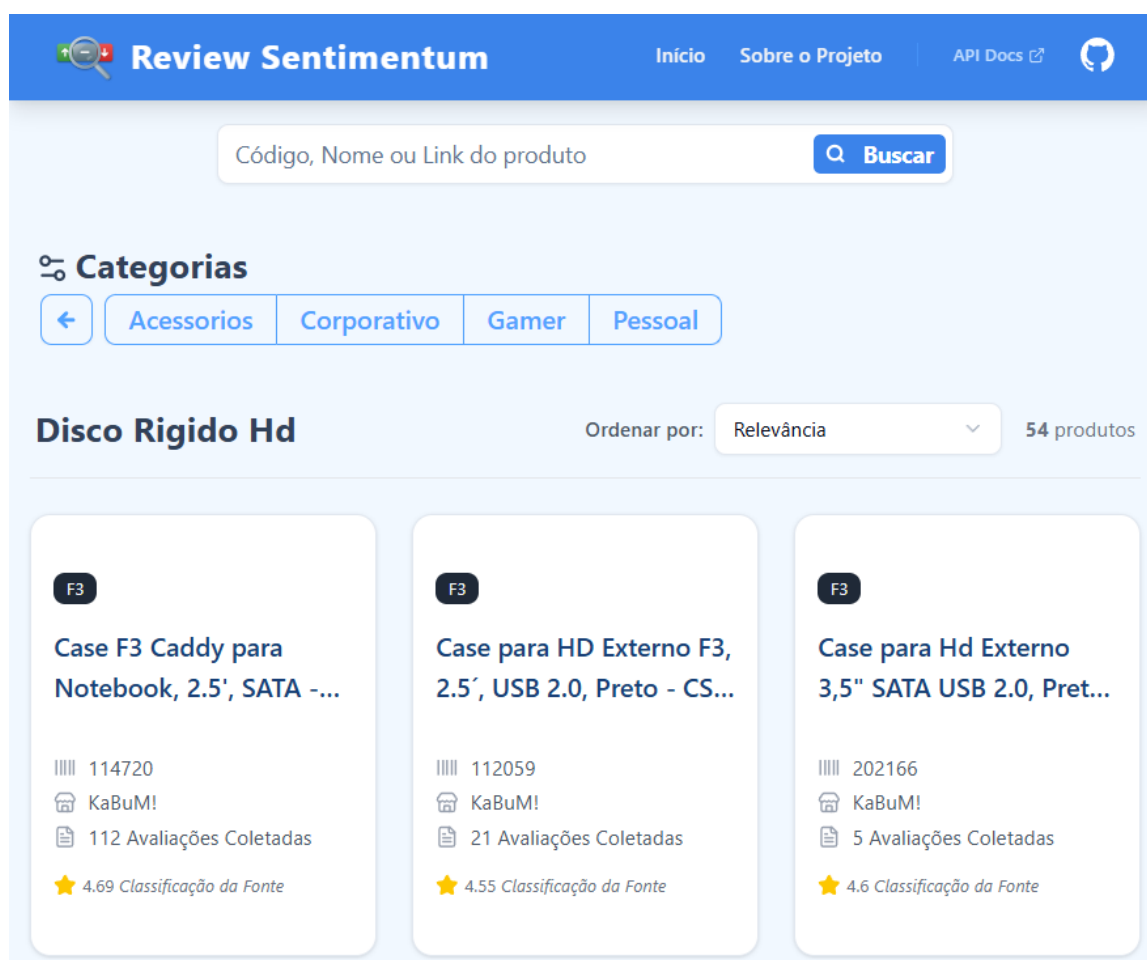
### 4.4 Desenvolvimento da Interface de Visualização

A interface de visualização do usuário, responsável por apresentar os dados e *insights* gerados, foi desenvolvida como uma aplicação web moderna utilizando o *framework* Next.js (Vercel, 2025), com a linguagem TypeScript para garantir a tipagem estática e a robustez dos dados manipulados. A arquitetura foi estruturada de forma modular e componentizada, empregando bibliotecas de componentes e CSS para construir uma interface responsiva, coesa e intuitiva. A comunicação com o *backend* é realizada por meio de requisições à API REST, permitindo a busca e a exibição dinâmica das informações de produtos, avaliações e as classificações de sentimentos. A aplicação adota uma abordagem de renderização mista, combinando componentes renderizados no servidor (*Server Components*) e no cliente (*Client Components*) para otimizar o desempenho e a interatividade.

O fluxo de interação do usuário foi expandido para oferecer múltiplas formas de descoberta de produtos, conforme ilustrado na Figura 2, que apresenta a página inicial do sistema. Esta interface atua como o centro de navegação, contendo um campo de entrada versátil capaz de interpretar buscas por nome, código do produto ou *link* direto, redirecionando o usuário

para a página de listagem ou diretamente para o detalhe do item, conforme o caso. Adicionalmente, foi implementado um menu lateral de categorias, visível na mesma figura, que permite ao usuário filtrar e visualizar todos os produtos de um segmento específico (ex: Placas de Vídeo, Processadores) organizados em *cards* com informações básicas. Também foi incluída uma seção informativa sobre o *Review Sentimentum*, explicando de forma simplificada o escopo acadêmico da ferramenta.

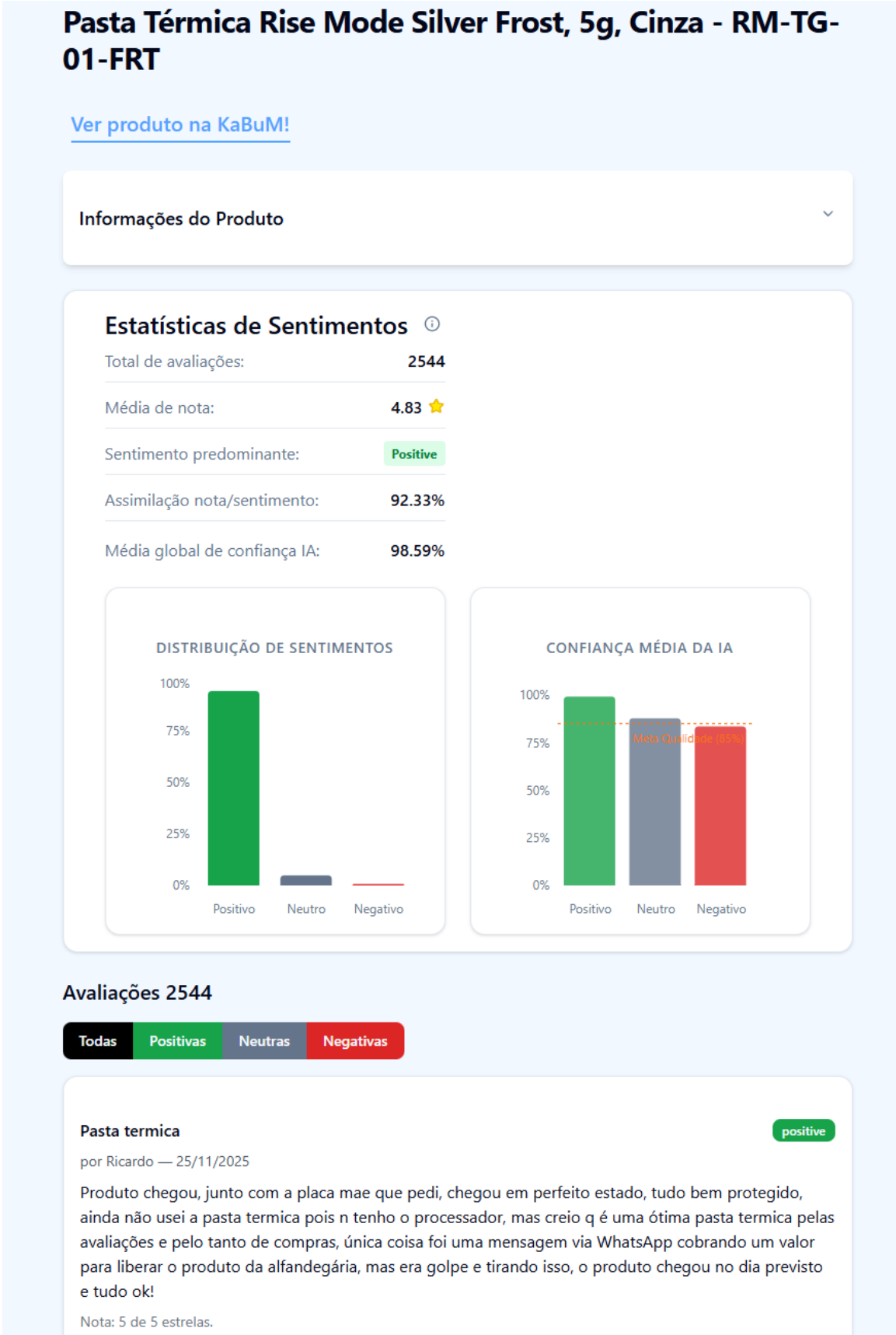
**Figura 2 – Captura de tela da página inicial com opções de busca e menu de categorias.**



**Fonte: Elaborado pelo autor.**

Por fim, ao selecionar um item nos resultados de busca ou na listagem de categorias, o usuário acessa a página de detalhes do produto (Figura 3), que exibe de forma consolidada os resultados da análise, apresentando gráficos com a distribuição dos sentimentos e os principais *insights*. Visando a clareza da informação para usuários não técnicos, foi incorporado nesta página um guia rápido de interpretação, instruindo o consumidor sobre o significado das métricas e indicadores de satisfação apresentados. Ainda nesta tela, o usuário pode filtrar as listagens de avaliações por polaridade (positiva, negativa ou neutra), facilitando a análise aprofundada e a compreensão do sentimento geral em relação ao produto, o que auxilia diretamente na tomada de decisão de compra.

Figura 3 – Página de detalhes do produto com a análise de sentimentos e *insights*.



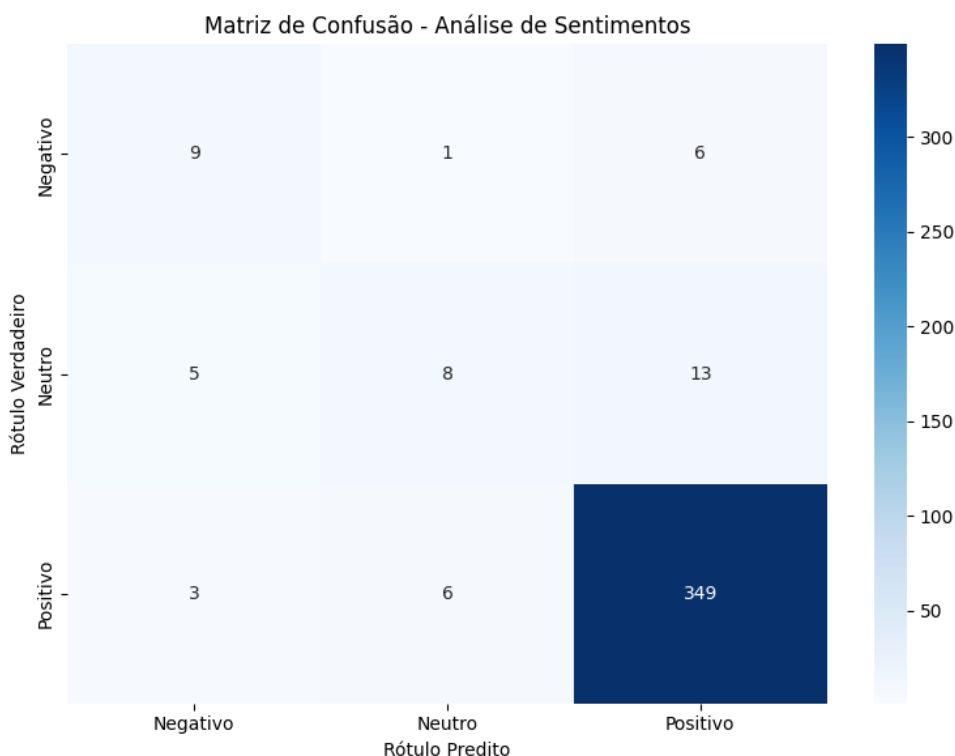


## 5 RESULTADOS

A execução do projeto resultou em um sistema plenamente funcional, cujas etapas geraram dados concretos que validam a eficácia da abordagem proposta. A fase de coleta de dados na plataforma Kabum culminou na extração bem-sucedida de um volume de 1.211 produtos de hardware com um total de 53.307 avaliações. O processo automatizado demandou um tempo total de aproximadamente 16 horas e 7 minutos, executando sem registros de falhas críticas ou interrupções, o que demonstra a robustez do script de *web scraping*. A coleção de texto obtida revelou-se um material rico e desafiador, caracterizado pela presença de linguagem técnica, informalidades, gírias e erros ortográficos, elementos inerentes ao nicho de informática.

Quanto ao desempenho do modelo de linguagem, o processo de *fine-tuning* alcançou métricas que validam a estratégia metodológica adotada. Nos testes com o conjunto de validação, o modelo obteve uma acurácia de 91,50% e um *F1-score* ponderado de 90,77%. A estabilidade do aprendizado foi corroborada pela métrica de *Cross-Entropy Loss* de 0,3603, evidenciando uma convergência consistente. Para uma análise granular dos acertos e erros, gerou-se a Matriz de Confusão apresentada na Figura 4.

**Figura 4 – Matriz de Confusão do modelo no conjunto de validação.**



Fonte: Elaborado pelo autor.

A análise da matriz e do relatório de classificação revela uma disparidade de desempenho correlacionada ao suporte de cada classe. A classe “positivo”, amplamente majoritária, apresentou resultados excelentes com precisão de 95% e revocação de 97% (*F1-score* de 96%).

Já as classes minoritárias impuseram maiores desafios: a classe “negativo” obteve um *F1-score* de 55%, enquanto a classe “neutro” atingiu 39%. Observa-se na matriz que a principal fonte de erro foi a classificação de itens neutros como positivos (13 ocorrências). Esse comportamento sugere que, no contexto de *e-commerce*, avaliações neutras frequentemente contêm termos positivos (ex: “Bom produto, mas caro”), o que, somado ao desbalanceamento dos dados, induz o modelo a uma tendência de positividade. Ainda assim, o resultado global confirma a aptidão do modelo para a vasta maioria dos casos reais.

Como resultado do desenvolvimento do *backend*, a API REST implementada com FastAPI demonstrou ser uma ponte de dados prática e eficiente. Sua arquitetura resultou em *end-points* robustos e uma estrutura de dados padronizada em formato JSON, de fácil consumo por aplicações cliente. A geração automática de uma documentação interativa contribuiu para a clareza e a praticidade de uso da API.

No *frontend*, o principal resultado é uma interface de visualização interativa, responsiva e intuitiva, focada em apresentar os *insights* da análise de forma clara ao consumidor. A aplicação web funcional permite ao usuário buscar de forma prática produtos de interesse, ver e examinar os dados consolidados por meio de elementos visuais, como gráficos que ilustram a distribuição percentual dos sentimentos. As avaliações são categorizadas e podem ser filtradas por polaridade, facilitando a análise aprofundada e a compreensão do sentimento geral dos compradores. A integração bem-sucedida de todas as etapas consolida o sistema como uma ferramenta eficaz de apoio à decisão de compra, transformando um grande volume de opiniões textuais em *insights* visuais e de fácil compreensão.

## 6 CONCLUSÃO

O presente trabalho alcançou seu objetivo principal ao desenvolver e implementar um sistema completo para coleta, análise e visualização de sentimentos em avaliações de produtos de informática. A solução proposta atua efetivamente como uma ferramenta de apoio à decisão, mitigando a sobrecarga de informações enfrentada pelos consumidores e transformando dados não estruturados em *insights* visuais e quantitativos. A estratégia metodológica de utilizar um modelo de linguagem base e submetê-lo a um processo de *fine-tuning* específico para o nicho de informática provou-se acertada. Os resultados obtidos, com uma acurácia global superior a 91%, demonstram que modelos especializados em domínios específicos podem oferecer desempenho robusto, compreendendo o vocabulário técnico e as nuances das avaliações de hardware. A integração deste modelo em uma arquitetura moderna, composta por *web scraping* automatizado, uma API REST em FastAPI e uma interface reativa, validou a viabilidade técnica da construção de soluções de ponta a ponta para o varejo digital.

Entretanto, a análise dos resultados também evidenciou limitações, particularmente no desempenho do modelo em classes minoritárias (neutras e negativas). Tal comportamento é reflexo direto do desbalanceamento natural das avaliações em plataformas de *e-commerce*. O

tempo disponível para o desenvolvimento restringiu a possibilidade de curadoria e rotulação de um *dataset* maior e mais equilibrado, o que poderia ter mitigado a confusão do modelo entre avaliações neutras e positivas. Portanto, conclui-se que, embora o modelo seja altamente eficaz para identificar a polaridade predominante, o refinamento da detecção de nuances exige um conjunto de dados de treinamento mais vasto e diversificado.

Como sugestão para trabalhos futuros, recomenda-se a expansão do *dataset* com técnicas de balanceamento sintético de dados. Além disso, sugere-se a integração de LLMs (*Large Language Models*) generativos modernos, fundamentados em arquiteturas como GPT (*Generative Pre-trained Transformer*). Embora a solução atual apresente a vantagem estratégica de operar localmente com baixo custo, a integração dessas tecnologias, ainda que implique maior custo computacional, permitiria avançar além da classificação de polaridade, possibilitando a geração automática de resumos textuais das avaliações e a extração de tópicos específicos (como desempenho, durabilidade ou custo-benefício), oferecendo ao usuário uma visão ainda mais detalhada e qualitativa sobre os produtos.

Por fim, este projeto contribui para a área de Sistemas de Informação ao demonstrar a aplicação prática de conceitos de Recuperação de Informação na Web e Engenharia de Dados, integrados a técnicas de Inteligência Artificial e Engenharia de Software. A solução desenvolvida exemplifica como a união destas disciplinas pode solucionar problemas reais de consumo, servindo como base para futuras investigações sobre a automação da análise de opinião no comércio eletrônico brasileiro.

## REFERÊNCIAS

- Adilmar Coelho Dantas. **caramelo-smile (Revision 2707a19)**. Hugging Face, 2024. Disponível em: <<https://huggingface.co/Adilmar/caramelo-smile-2>>.
- ALEKSENDRIĆ, M. *et al.* **Full Stack FastAPI, React, and MongoDB: Fast-Paced Web App Development with the FARM Stack**. 2. ed. Birmingham: Packt Publishing, 2024. Acesso em: 02 out. 2025. ISBN 9781835886762. Disponível em: <<https://www.oreilly.com/library/view/full-stack-fastapi/9781835886762/>>.
- FILHO, S. S. M.; LIMONGI, R.; MARTINS, F. S. Análise de sentimento: tutorial com python. **ReMark - Revista Brasileira de Marketing**, v. 24, n. 3, p. e26739, ago. 2025. Disponível em: <<https://periodicos.uninove.br/remark/article/view/26739>>.
- Hugging Face. **Transformers Documentation**. 2025. Acesso em: 13 out. 2025. Disponível em: <<https://huggingface.co/docs/transformers/pt/index>>.
- LIMA, W. H. d. **tcc\_sentiment-analysis-ecommerce: Código-fonte para o Sistema de Análise de Sentimentos para Avaliações de Produtos de E-commerce**. GitHub, 2025. Acesso em: 14 set. 2025. Disponível em: <<https://github.com/ICEI-PUCMinas-PSG-SI-TI/psg-si-2025-2-p8-tcc-analisesentimentos>>.
- LOUREIRO, D. *et al.* TimeLMs: Diachronic language models from Twitter. In: **Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations**. Dublin, Ireland: Association for Computational Linguistics, 2022. p. 251–260. Disponível em: <<https://aclanthology.org/2022.acl-demo.25>>.
- MITCHELL, R. **Web Scraping com Python**. 3. ed. O'Reilly Media, Inc., 2025. Acesso em: 02 set. 2025. ISBN 9798341638013. Disponível em: <<https://learning.oreilly.com/library/view/web-scraping-com/9798341638013/>>.
- MongoDB, Inc. **PyMongo Driver - MongoDB Docs**. 2025. Acesso em: 12 out. 2025. Disponível em: <<https://www.mongodb.com/pt-br/docs/languages/python/pymongo-driver/current/>>.
- RAMÍREZ, S. **FastAPI**. 2025. Acesso em: 15 out. 2025. Disponível em: <<https://fastapi.tiangolo.com/>>.
- RODRIGUES, M. B.; FLÁVIO, V. V. S.; NETO, G. H. Desenvolvimento e implementação do workflow para realizar a análise de sentimentos como uma possível estratégia comercial. **REVISTA ELETRÔNICA DE COMPUTAÇÃO APLICADA**, Departamento de Computação do Uni-FACEF, v. 2, n. 1, 2021. ISSN 2675-8830. Disponível em: <<http://periodicos.unifacef.com.br/reca/article/view/2300>>.
- SEBRAE. **61% dos brasileiros preferem e-commerce ao varejo físico**. 2023. Acesso em: 15 out. 2025. Disponível em: <<https://sebrae.com.br/sites/PortalSebrae/61-dos-brasileiros-preferem-e-commerce-ao-varejo-fisico,d6a8dd3257c67810VgnVCM1000001b00320aRCRD>>.
- Selenium Project. **O Projeto Selenium de Automação de Navegadores [Documentação]**. 2025. <<https://www.selenium.dev/pt-br/documentation/>>. Acesso em: 12 out. 2025.
- SILVA, D. *et al.* Modelos transformers para a análise automática de satisfação na plataforma consumidor.gov.br. In: **Anais do XII Workshop de Computação Aplicada em Governo Eletrônico**. Porto Alegre, RS, Brasil: SBC, 2024. p. 1–13. ISSN 2763-8723. Disponível em: <<https://sol.sbc.org.br/index.php/wcge/article/view/29521>>.

SILVA, P. F. d. *et al.* Sistema de recomendação em e-commerce com modelos de llm e computação em nuvem: um relato de experiência. **REVISTA BRASILEIRA MILITAR DE CIÊNCIAS**, v. 11, n. 25, jul. 2025. Disponível em: <<https://rbmc.org.br/rbmc/article/view/198>>.

SOUZA, W. L. *et al.* Sistema de análise de sentimento em comentários relacionados a universidades de ensino superior em redes sociais (venera). **REVISTA DO FÓRUM GERENCIAL**, Centro Universitário de Patos de Minas (UNIPAM), Patos de Minas, v. 1, n. 2, 2021. Disponível em: <<https://revistas.unipam.edu.br/index.php/forumgerencial/article/view/2430>>.

Vercel. **Next.js Docs**. 2025. Acesso em: 15 out. 2025. Disponível em: <<https://nextjs.org/docs>>.