Equipe AdaLove

Fatec São José dos Campos - Professor Jessen Vidal

Documentação Técnica - Aprendizagem por Projetos Integrados

AdaTech

Índice

1. Visão Geral 2
2. Arquitetura do Sistema 3
3. Tecnologias Utilizadas 5
4. Modelagem de Dados 7
5. API / EndPoints 9
15. Considerações Finais 9
16. Glossário 10

Esta documentação destina-se a desenvolvedores, professores, e todos os envolvidos no desenvolvimento do sistema AdaTech.

Equipe AdaLove

Fatec São José dos Campos - Professor Jessen Vidal

1. Visão Geral

AdaTech é um sistema web que automatiza a geração da instrução de importação para uso do despachante aduaneiro. Ele foi desenvolvido para atender à empresa cliente TecSys, que anteriormente buscava por esses dados de forma manual.

Para cada item importado, a instrução precisa conter os seguintes dados obrigatórios:

Part number* (código interno do componente);

Descrição técnica do item (espécie, marca, modelo, etc.);

Classificação fiscal (NCM) do produto;

Nome e endereço do fabricante, incluindo o país de origem.

Essas informações são exigidas pela Declaração de Importação (DI)*, especialmente nos campos de “classificação fiscal da mercadoria” e “identificação da origem”.

Cada código NCM está associado a alíquotas* específicas de impostos. O erro de classificação pode gerar multas, pagamento indevido de tributos ou até a retenção da carga na alfândega.

Por isso, o AdaTech é projetado para gerar descrições técnicas claras e completas, com o objetivo de garantir uma identificação precisa do item, conformidade legal e agilidade no despacho.

1.1 Funcionamento do sistema

O processo se inicia com o envio, pelo usuário, de uma lista de itens — como, um PDF de ordem de compra.

A partir disso, o sistema executa as seguintes etapas:

- **Extração de dados**: um módulo de processamento de documentos analisa o arquivo enviado e extrai os campos relevantes (como part number, fabricante e descrição inicial);

- **Consulta à base local:** com base no part number o sistema localiza e retorna uma descrição técnica padronizada do item;

- **Busca do fabricante:** por meio de técnicas de web scraping*, o sistema localiza informações atualizadas sobre o fabricante em sites relevantes;

- **Classificação fiscal (NCM):** a IA analisa as características técnicas do item e sugere o código NCM mais adequado;

- **Geração da instrução aduaneira:** todas as informações coletadas são organizadas em um formulário. Antes da exportação final, o usuário pode revisar e ajustar os dados manualmente, assegurando a conformidade com as exigências legais.

Ao final, o sistema gera a planilha estruturada nos moldes exigidos que será anexada à Declaração de Importação. Essa planilha inclui a "Especificação da Mercadoria", campo obrigatório que deve conter uma descrição completa, em português, com todos os dados necessários à correta identificação comercial e a classificação fiscal do produto.

O sistema AdaTech automatiza esse processo de ponta a ponta, reduzindo erros, evitando penalidades e acelerando o tempo de liberação aduaneira.

2. Arquitetura do Sistema

2.1 Componentes principais e responsabilidades

Frontend (React):

- Upload de arquivos PDF e requisições HTTP para o backend.

- Exibição de resultados retornados em Excel.

Backend (FastAPI):

- Expor endpoints REST para receber PDFs e retornar resultados.

- Orquestrar o processamento usando pipeline_service.

- Integrar serviços de extração de texto, RAG e IA.

Services:

- pipeline_service.py — coordena todo o fluxo:

- Recebe PDF.

- Extrai texto → part numbers → descrições.

- Busca na base NCM usando rag_service.

- Consulta IA (Qwen3) via llm_service.

- Monta arquivo Excel com resultado final.

- pdf_service.py — responsável por abrir o PDF, extrair texto e identificar dados relevantes.

- rag_service.py — busca de NCM usando TF-IDF + Similaridade de Cosseno sobre a base ncm.csv carregada em memória.

- llm_service.py — integra com o modelo Qwen3 via Ollama, aplica RAG para melhorar respostas.

Modelos / Dados:

- Base NCM carregada em cache para performance.

- CSV ncm.csv contém colunas: ncm; descricao; codificação: latin1; separador: vírgula

2.3 Padrões Arquiteturais adotados

MVC (Model-View-Controller):

Controller → routes/process_pdf.py (rotas API).

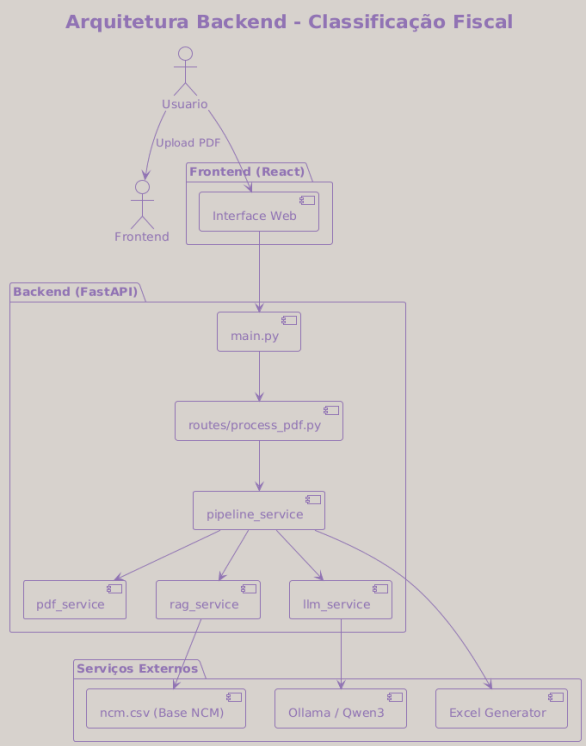
Model → Representação dos dados extraídos e resultados.

View → JSON de resposta ou arquivo Excel.

2.5 Motivações para escolha dos padrões

A separação em camadas (MVC) resulta em código mais organizado, manutenível e testável, facilitando a colaboração da equipe de desenvolvimento e tornando a aplicação mais escalável e flexível.

2.6 Diagrama da Arquitetura



3. Tecnologias Utilizadas

3.1 Linguagens de Programação:

Python → escolhido para o backend pela ampla disponibilidade de bibliotecas para processamento de texto, integração com IA e geração de arquivos Excel. Além disso, o ecossistema de frameworks web (FastAPI) é bem suportado.

TypeScript → usado no frontend (com React) para trazer segurança no desenvolvimento, fornecendo tipagem estática, melhor manutenção do código e prevenção de erros em tempo de compilação.

3.2 Frameworks e Bibliotecas:

FastAPI → framework backend em Python, escolhido pela performance, facilidade de uso e documentação automática integrada via Swagger e Redoc.

React → biblioteca para construção de interfaces modernas e reativas, que permite separar responsabilidades e facilita a integração com APIs REST.

Uvicorn → servidor ASGI rápido e eficiente, utilizado para rodar o FastAPI em produção.

pandas / openpyxl → bibliotecas usadas para manipulação de dados e geração de relatórios em Excel.

scikit-learn → utilizado para aplicar TF-IDF e Similaridade de Cosseno na busca de NCMs.

3.3 Banco de Dados:

PostgreSQL (planejado) → banco de dados relacional robusto, escalável e confiável, ideal para armazenar histórico de consultas, logs e resultados futuros do sistema. Atualmente, a solução lê dados de um CSV (`ncm.csv`) carregado em cache, mas a migração para PostgreSQL permitirá melhor escalabilidade e consultas mais complexas.

3.4 Outras Ferramentas:

Ollama → plataforma para rodar modelos de linguagem localmente, garantindo privacidade, baixo custo e independência de serviços externos.

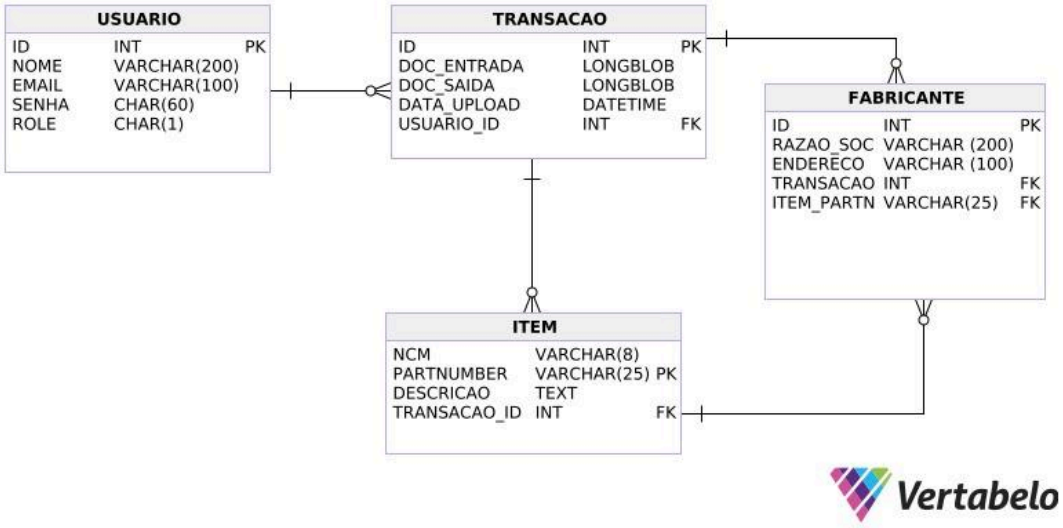
Qwen3 (via Ollama) → modelo de linguagem utilizado para sugerir classificações fiscais com base no contexto extraído e nos resultados da busca RAG.

Swagger UI → interface gerada automaticamente pelo FastAPI para testar endpoints de forma prática.

Redoc → documentação detalhada da API, também gerada automaticamente pelo FastAPI.

4. Modelagem de Dados

4.1 Modelagem física - AdaTech



4.2 Entidades principais e seus relacionamentos

Usuário: Representa a entidade que realiza as ações no sistema.

Transação: Representa a entidade que consolida a submissão do pedido de compra, e a devolução do documento de instrução aduaneira.

Item: Representa cada item aduaneiro detectado no pedido de compra, ele é submetido através da transação, e possui as informações, como partnumber, ncm, etc.

Fabricante: Representa os fabricantes salvos no banco de dados, que podem ser selecionados na edição do formulário, se for necessário.

5. API / EndPoints

A API foi construída com FastAPI e expõe endpoints REST para processamento de documentos. Atualmente, está disponível o seguinte recurso:

5.1. /process_pdf

Método: **POST**

Descrição: Recebe um arquivo PDF com descrições de componentes eletrônicos, realiza extração de informações (PartNumbers e descrições), consulta a base de NCM via RAG + LLM (Qwen3/Ollama), enriquece os dados com informações de fabricante/localização (via scraper) e retorna um arquivo Excel (.xlsx) contendo a classificação fiscal sugerida.

5.2. Parâmetros da Requisição

file: **(obrigatório)**

Tipo: UploadFile

Formato: application/pdf

Enviado no corpo da requisição como multipart/form-data.

5.3. Respostas

5.3.1. 200 OK

Retorna um arquivo Excel (application/vnd.openxmlformats-officedocument.spreadsheetml.sheet) com as colunas:
partnumber → código do componente.
fabricante → fabricante identificado.
localizacao → país ou região de fabricação (quando disponível).
ncm → código NCM sugerido.
descricao → descrição fiscal associada ao NCM.

5.3.2. 400 Bad Request

"Envie um arquivo PDF." (caso o arquivo não seja PDF).

"Arquivo vazio." (caso o arquivo não contenha dados).

"Nenhum item encontrado no PDF." (caso não tenha extração válida).

5.3.3. 500 Internal Server Error

Erro extraindo PDF: ..." (problema na leitura do arquivo).

"Erro RAG" (falha no mecanismo de recuperação/classificação).

Exemplo de Requisição

cURL:

```
curl -X POST "http://localhost:8000/process_pdf" \
  -H "accept: application/vnd.openxmlformats-officedocument.spreadsheetml.sheet" \
  -F "file=@lista_componentes.pdf"
```

15. Considerações Finais

15.1 Pontos de Melhoria

Otimizar o desempenho do RAG usando embeddings persistentes em um vetor database(ex.: Milvus, FAISS, ChromaDB), em vez de TF-IDF.

Aumentar a robustez do processo de normalização, tratando mais variações de descrições de componentes.

Implementar conexão com banco de dados PostgreSQL para armazenar fabricantes

Adicionar o login de usuário, para permitir acesso ao histórico de itens enviados.

15.2 Lições Aprendidas

O uso de RAG com CSV mostrou-se uma solução rápida e eficaz, mas tem limitações em escalabilidade e precisão.

Ollama simplificou a execução de modelos locais, mas exige máquina robusta e ajustes de memória.

A arquitetura baseada em serviços separados (pipeline_service, pdf_service, rag_service, etc.) facilitou a manutenção e a substituição de partes do sistema sem impacto no restante.

A documentação automática do FastAPI (Swagger/Redoc) acelerou os testes e validação dos endpoints.

16. Glossário

PartNumber: Código identificador único de um componente eletrônico.

NCM (Nomenclatura Comum do Mercosul): Código de classificação fiscal usado para identificar mercadorias em transações internacionais.

RAG (Retrieval-Augmented Generation): Técnica que combina busca em base de dados com geração de linguagem natural por LLMs.

TF-IDF (Term Frequency–Inverse Document Frequency): Técnica estatística para identificar a relevância de palavras em textos.

LLM (Large Language Model): Modelo de linguagem de grande porte, como o Qwen3, usado para processamento de linguagem natural.

Ollama: Ferramenta para rodar modelos de IA localmente.

FastAPI: Framework Python para construção de APIs rápidas e escaláveis.

Swagger/Redoc: Interfaces gráficas de documentação automática geradas pelo FastAPI.