



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. Normalização de Dados	9
6. API / EndPoints	10
7. Considerações Finais	13
8. Glossário	14

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 um processo de login e/ou cadastro do usuário, logo após o usuário realiza o envio 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;



Equipe AdaLove

Fatec São José dos Campos – Professor Jessen Vidal

– **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.



Equipe AdaLove

Fatec São José dos Campos – Professor Jessen Vidal

- 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

O frontend se comunica com o backend (FastAPI) via API REST.

2.3 Padrões Arquiteturais adotados

MVC (Model-View-Controller):

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

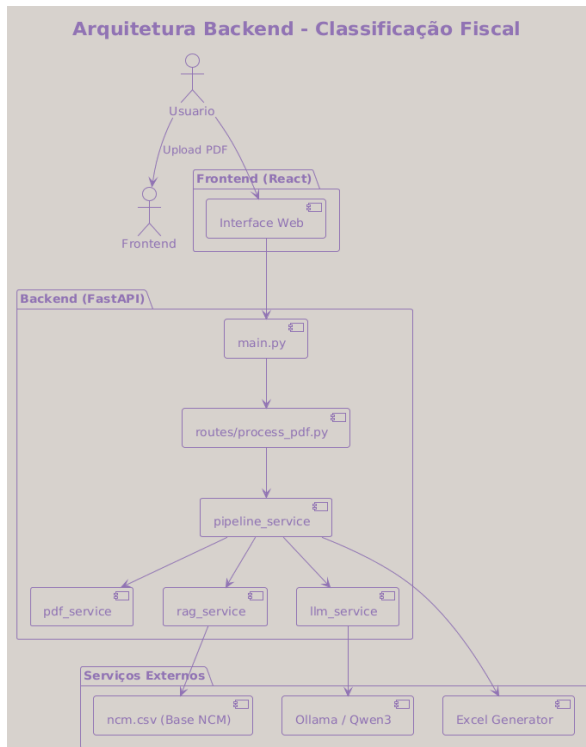
Model → Representação dos dados extraídos e resultados.

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.



Equipe AdaLove

Fatec São José dos Campos – Professor Jessen Vidal

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 → banco de dados relacional robusto, escalável e confiável, ideal para armazenar histórico de consultas, logs e resultados futuros do sistema.

3.4 Outras Ferramentas:

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

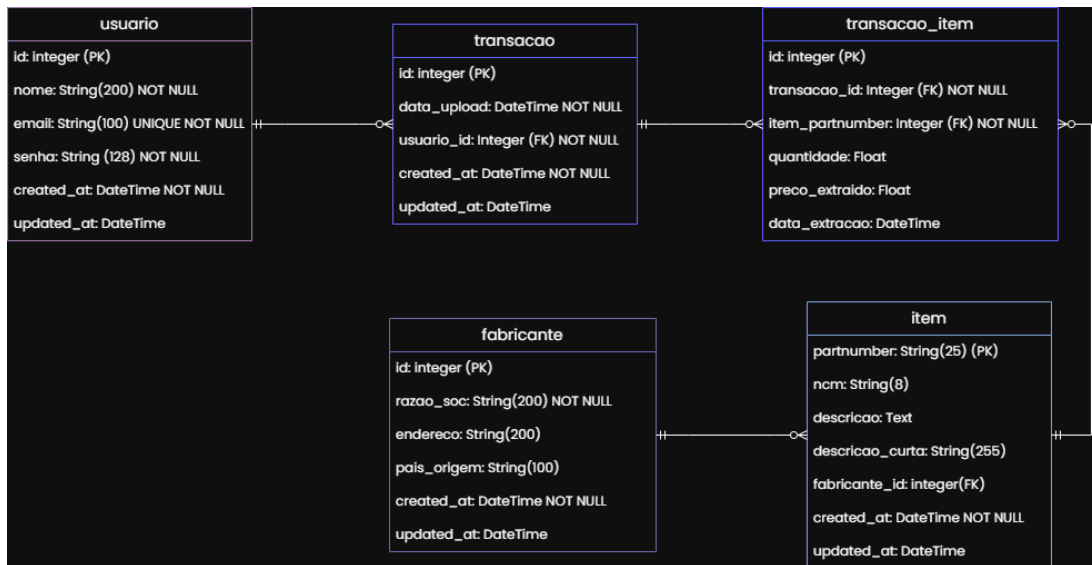
Qwen3 (via Ollama) → modelo de linguagem utilizado para sugerir classificações fiscais com base no contexto extraído e nos resultados da busca RAG.

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

Usuário: Armazena informações dos usuários do sistema.

Campos:

- - id: Primary Key, Integer
- - nome: String(200), obrigatório
- - email: String(100), único, obrigatório
- - senha: String(128), obrigatório (hash)
- - created_at: DateTime, timestamp de criação
- - updated_at: DateTime, timestamp de atualização

Relacionamentos: 1:N com Transacao (um usuário pode ter várias transações)



Equipe AdaLove

Fatec São José dos Campos – Professor Jessen Vidal

Transação: Representa uma transação de um usuário, associada a um item do documento.

Campos:

- - id: Primary Key, Integer
- - data_upload: DateTime, timestamp da transação
- - usuario_id: ForeignKey para Usuario.id, obrigatório
- - item_partnumber: ForeignKey para Item.partnumber, obrigatório
- - created_at: DateTime, timestamp de criação
- - updated_at: DateTime, timestamp de atualização

Relacionamentos: N:1 com Item; N:1 com Usuario

Item: Representa um item/produto extraído do documento.

Campos:

- - partnumber: Primary Key, String(25)
- - ncm: String(8)
- - descricao: Text, descrição longa
- - descricao_curta: String(255), descrição curta
- - fabricante_id: ForeignKey para Fabricante.id, opcional
- - created_at: DateTime, timestamp de criação
- - updated_at: DateTime, timestamp de atualização

Relacionamentos: 1:N com Transacao; N:1 com Fabricante (opcional)

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

5. Normalização — Aplicando 1FN, 2FN, 3FN

5.1. 1FN – Primeira Forma Normal

Regras:

- - Todos os dados atômicos (sem listas dentro de células)
- - Sem colunas repetidas
- - Cada registro identificável por chave primária

Problemas comuns:

- - "info_adicional" contendo listas → não atômico
- - Descrições longas misturando detalhes diferentes
- - Múltiplos produtos do mesmo arquivo duplicando "data_transacao"

Correções:

- - Criar tabela Transacao
- - Criar tabela Item
- - Remover campos compostos

5.2. 2FN – Segunda Forma Normal

Regras:

- - Estar na 1FN
- - Todo atributo precisa depender da chave inteira, não apenas parte dela

Problemas comuns:

- - Se a chave fosse composta (ex.: id_transacao + item), campos como fabricante, país de origem e NCM não dependem do item → gera redundância.

Correções:

- - Criar tabelas próprias para entidades independentes: como Fabricante



Equipe AdaLove

Fatec São José dos Campos – Professor Jessen Vidal

5.3. 3FN – Terceira Forma Normal

Regras:

- - Estar na 2FN
- - Nenhum atributo depende de outro atributo não-chave (dependência transitiva)

Exemplos de Problemas comuns:

- - “pais_origem do fabricante” depende de “fabricante” e não da chave primária

Correções:

- - Os dados do fabricante vão para tabela própria do fabricante.

5.4. Benefícios dessa normalização

- - Redução drástica de redundância.
- - Evita inconsistências.
- - Melhora performance em consultas
- - Facilita atualização de informações externas (scraper → fabricante/NCM)

6. API / EndPoints

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

6.1. EndPoints de Autenticação

6.1.1. /auth/login

Método: POST

Descrição: Endpoint de login do usuário



Equipe AdaLove

Fatec São José dos Campos – Professor Jessen Vidal

6.1.2. /user/profile

Método: GET

Descrição: Endpoint que puxa o perfil do usuário

6.2. EndPoints do PDF

6.2.1. /extract_from_pdf

Método: POST

Descrição: Recebe um arquivo PDF com descrições de componentes eletrônicos, realiza extração de informações.

6.2.2. /process_items

Método: POST

Descrição: Processa uma lista de itens e informações extraídas.

Para cada item realiza uma busca (Web scraping*) com o objetivo de encontrar o fabricante daquele produto, o NCM e gerar duas descrições, uma curta e uma mais longa.

6.2.3. /generate_excel

Método: POST

Descrição: Gera o arquivo com extensão .XLSX (formato Excel) com os itens já processados.

6.2.4. /save_items

Método: POST



Equipe AdaLove

Fatec São José dos Campos – Professor Jessen Vidal

Descrição: Salva ou atualiza os itens já processados no banco de dados, vinculando ao usuário e à Transação.

6.3. EndPoints de Teste

6.3.1. /test_pdf

Método: POST

Descrição: EndPoint de teste do PDF, cria um DataFrame de teste e gera um Excel em memória.

6.4. EndPoints do Usuário

6.4.1. /users

Método: POST

Descrição: EndPoint que cria o usuário e recebe o nome, email e senha, que serão usados para realizar login na aplicação.

6.5. EndPoints de Documentação

6.5.1. /docs

Descrição: Endpoint de documentação do software

6.6. Respostas

200: OK

201: Created



Equipe AdaLove

Fatec São José dos Campos – Professor Jessen Vidal

400: Bad Request

/test_pdf: "Envie um arquivo PDF válido." (caso o arquivo não seja PDF).

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

/extract_from_pdf: "Envie um arquivo PDF válido com nome." (caso o arquivo não seja PDF).

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

/process_items: "Lista de itens para processar está vazia." (caso não tenha dados válidos).]

/generate_excel: "Nenhum item fornecido para gerar o Excel." (caso não tenha dados válidos).

/save_items: " Nenhum item fornecido."

401: Unauthorized

/auth/login: "E-mail e/ou senha incorretos."

500: Internal Server Error

/extract_from_pdf: "Erro durante a extração do PDF" (problema na leitura do arquivo).

/generate_excel: "Erro interno ao gerar o arquivo Excel." (problema na geração do .xlsx).

/save_items: " Erro interno ao salvar."

7. Considerações Finais

7.1 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.



Equipe AdaLove

Fatec São José dos Campos – Professor Jessen Vidal

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

8. 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.

Web scraping: Forma de mineração que permite a extração de dados de sites da web, convertendo-os em informação estruturada para posterior análise.

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.