

Tech Challenge - Projeto de API - Vitibrasil

ERIC LOPES MELLO

Vídeo de Apresentação: https://drive.google.com/file/d/1Cw7zF15bDPE4DL8y_zIpwsu5R1K5aaE/view?usp=sharing

Descrição Geral:

O projeto **Vitibrasil** é uma aplicação web desenvolvida em **Flask** que centraliza e gerencia dados sobre a produção, importação, exportação, comercialização e processamento de produtos de diversos países. Ela permite que os usuários autenticados façam o **download** de arquivos CSV relacionados a essas métricas, visualizem os dados em forma de **tabelas HTML**, e gerem **gráficos de análise** para comparar os dados de diferentes países. O projeto também inclui a funcionalidade de geração dinâmica do arquivo "Dimensão País", que consolida as informações em um formato organizado para facilitar análises mais detalhadas posteriores. O sistema permitirá a realização de análise e previsão de importações e exportações de uvas e seus derivados por meio de técnicas de Machine Learning, assim como a visualização de tendências e padrões.

Acesso: A aplicação pode ser acessada no navegador através de <https://vitibrasil.onrender.com/login>

- **Login:** user1
- **Senha:** password1

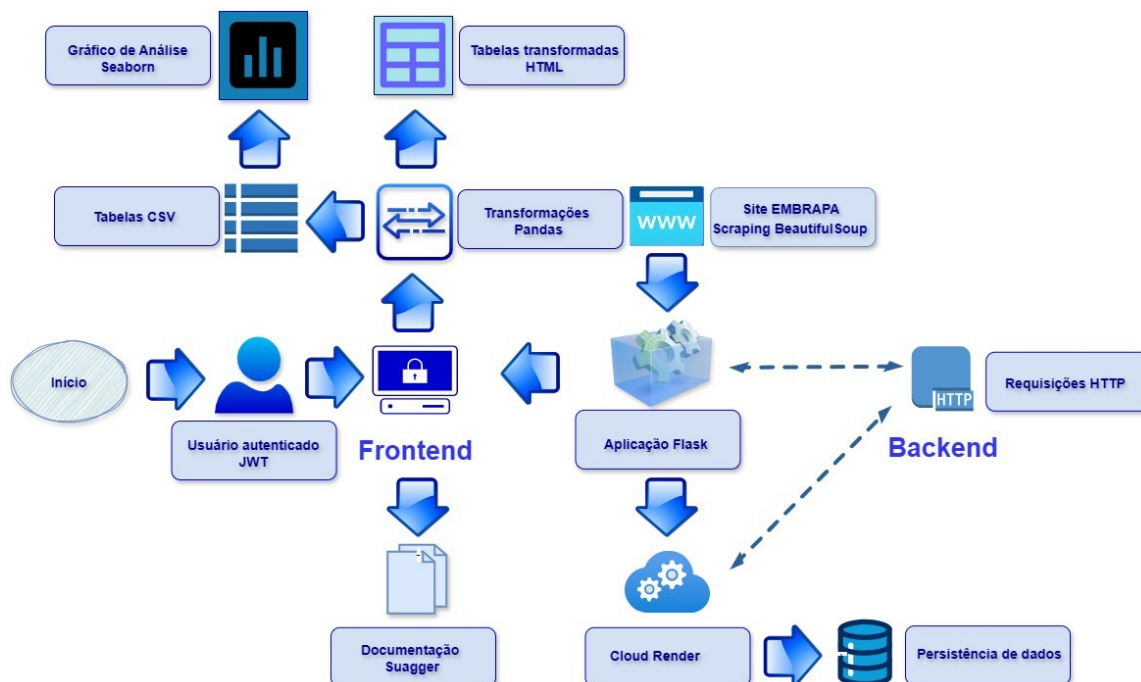
A documentação Swagger da API está disponível em <https://vitibrasil.onrender.com/apidocs>

O código fonte do projeto será armazenado no repositório <https://github.com/ericlmello/vitibrasil> do GIT, conectado ao Render. Para acesso local, deve-se executar o script do arquivo `app.py` e acessar a url local <http://127.0.0.1:5000/login>

Desenho da Arquitetura do Projeto:


Ele apresenta as camadas principais: Frontend, Backend, Camada de Negócios, e Persistência, além de futuras integrações como o Data Lake automatizado, dashboards e modelos de machine learning para previsão de tendências e análise de anomalias.

Arquitetura Geral do Projeto:



1. Camada de Apresentação (Front-end)

- **Login Page (login.html):** Página HTML para que os usuários forneçam suas credenciais. O design deve ser simples e intuitivo, com validações básicas no front-end e suporte a envio de tokens JWT para autenticação.
- **Visualização de Tabelas:** As tabelas HTML exibem os dados dos arquivos CSV importados, facilitando a análise dos dados de produção, exportação e importação. O DataTables pode ser usado para melhorar a interatividade (paginação, busca, etc.).
- **Gráficos:** Gráficos dinâmicos gerados a partir dos dados importados (importação/exportação/produção) usando bibliotecas como Chart.js ou D3.js para visualização intuitiva e interativa.
- **Autenticação/Security:** A autenticação é gerenciada via JWT (JSON Web Tokens). O token JWT é incluído no cabeçalho das requisições para endpoints protegidos, garantindo que apenas usuários autenticados possam acessar determinadas funcionalidades.



Vitibrazil - Viticultura Brasileira

Usuário:

user1

Senha:

Login

Baixar Arquivos

Baixar Producao

Baixar Processamento

Baixar Comercializacao

Baixar Importacao

Baixar Exportacao

Baixar Dimensao Pais para Analise

Visualizar Arquivos CSV

Visualizar Producao

Visualizar Processamento

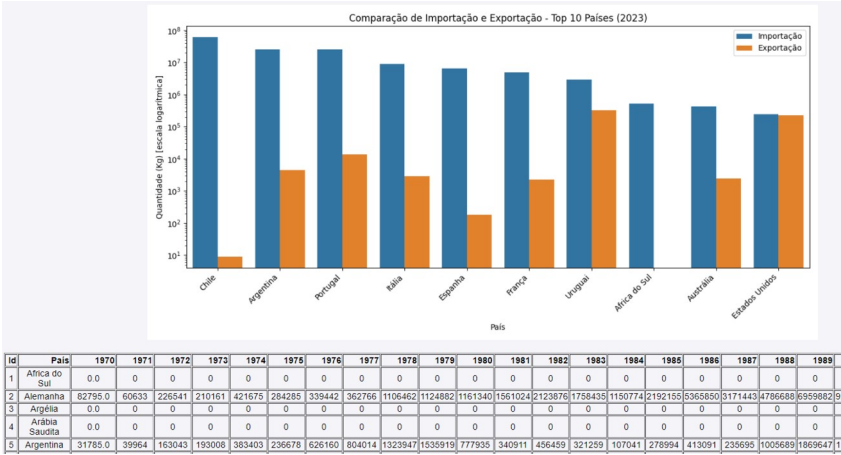
Visualizar Comercializacao

Visualizar Importacao

Visualizar Exportacao

Visualizar Gráfico de Análise

Visualizar Gráfico de Análise



Frontend

2. Camada de Aplicação (Back-end)

- **Ingestão de Dados (download.py):** Responsável por fazer o scraping dos dados do site da Embrapa, que oferece informações sobre Produção, Processamento, Comercialização, Importação e Exportação. Utiliza a biblioteca BeautifulSoup para coletar os dados e armazená-los em arquivos CSV.
- **Flask:** O framework Flask gerencia as rotas HTTP da aplicação. Ele lida com requisições para:
 - Exibição dos dados CSV;
 - Geração de gráficos com base nos dados agregados;
 - Download de arquivos CSV;
 - Documentação e teste de endpoints via Swagger.
- **Autenticação JWT:** Garante que endpoints protegidos sejam acessíveis somente a usuários autenticados. Tokens JWT são gerados durante o login e precisam ser incluídos nas requisições subsequentes.
- **Swagger:** A documentação dos endpoints é feita com Swagger, permitindo que os usuários visualizem e testem as rotas da API.
- **Manipulação de CSV:** A aplicação manipula arquivos CSV para gerar tabelas HTML e gráficos, utilizando Pandas para processar e agregar os dados.

3. Camada de Negócio

- **Processamento dos Dados (config.py e app.py):**
 - **Armazenamento:** Os arquivos CSV coletados são armazenados localmente em um diretório específico (ex.: assets).
 - **Transformação:** Utilizando Pandas, os dados são pré-processados (limpeza, soma de valores semestrais, remoção de duplicatas e nulos) para facilitar a análise.
 - **Funções:** O arquivo app.py contém a função que gera gráficos de barras comparativos dos 10 principais países que importam e exportam para o Brasil no último ano. Essa funcionalidade será eventualmente transferida para um novo arquivo focado em Machine Learning.
 - **Rotas:** Também no app.py, são definidas as rotas que mapeiam URLs para funções específicas da aplicação (ex.: visualização de dados, download de arquivos, geração de gráficos).
- **Processamento de Dados:** Os dados dos arquivos CSV relacionados a importação, exportação, produção, comercialização e processamento são agregados com Pandas. A aplicação então gera relatórios (CSV) e gráficos que ajudam na análise dos principais países exportadores e importadores.
- Ao arquivo app.py foi acrescentada a seguinte linha de código para garantir a **atualização** do gráfico em caso de acréscimo de anos posteriores:

```
# Encontrar o último ano presente nas colunas
anos_disponiveis = [col for col in df.columns if col.isdigit()] ultimo_ano = max(anos_disponiveis)
```
- **Criação de Dimensão País (utils.py):** Script que processa e transforma os dados para análise. Por exemplo, somando valores semestrais de cada ano e eliminando duplicatas e valores nulos.

4. Camada de Persistência (Armazenamento de Arquivos)

- **Diretório de Armazenamento de Arquivos CSV:** Todos os arquivos CSV relacionados à importação, exportação,

produção, etc. são salvos localmente em um diretório específico (ex.: assets).

- **Static Directory:** Diretório onde os gráficos gerados são armazenados (ex.: static/graphs) para serem disponibilizados para visualização pelos usuários.

5. Futuras Integrações

- **Data Lake Automatizado:** Planejamento para integração com fontes de dados externas. A ideia é automatizar a ingestão de novos dados de múltiplas fontes, mantendo o sistema atualizado.
- **Dashboards:** Integração futura com um dashboard interativo que permita visualização em tempo real dos dados, oferecendo filtros e opções de personalização.
- **Alimentação do Modelo de Machine Learning:**
 - **Modelo de ML:** Um modelo de Machine Learning será treinado com os dados CSV pré-processados. O modelo poderá ser usado para previsão de tendências e detecção de anomalias nos dados, utilizando frameworks como Scikit-learn para o treinamento e integração com a aplicação.
 - **Atualização do Modelo:** O modelo de ML será atualizado conforme novos dados forem carregados no sistema, podendo operar em tempo real ou em ciclos periódicos, conforme a necessidade.

Estrutura de diretórios:

- **app.py:** Principal arquivo da aplicação que define as rotas HTTP, gerencia a autenticação JWT, e coordena a interação entre as demais funcionalidades.
- **config.py:** Define configurações globais da aplicação, como o diretório de armazenamento de arquivos CSV.
- **auth.py:** Contém as funções de login e autenticação dos usuários.
- **download.py:** Gerencia o download dos arquivos CSV e a visualização de tabelas HTML.
- **utils.py:** Contém funções utilitárias, como a geração do arquivo "Dimensão País" e correções de CSV.
- **templates/:** Diretório que contém os arquivos HTML utilizados na renderização das páginas (e.g., login.html).
- **static/:** Diretório onde gráficos e outros arquivos gerados são armazenados.

Plano de Deploy:

Objetivo: Implementar um processo de entrega contínua para o projeto, utilizando o **Render** como servidor para hospedar a aplicação, permitindo a atualização frequente e segura da aplicação, garantindo que os usuários tenham acesso às funcionalidades mais recentes sem interrupções.

1. Configuração de Build e Deploy:

```
pip install -r requirements.txt (instalação de dependências)
```

```
gunicorn app:app --bind 0.0.0.0:8000 (comando de inicialização do Flask no Render)
```

2. Configuração de Deploy Automático: Ativação da opção de deploy automático sempre que houver um novo commit no branch principal (após novos testes). Isso garantirá que cada nova atualização no código-fonte seja implantada automaticamente.

3. Monitoramento e Manutenção: Utiliza-se o painel do Render para monitorar o status da aplicação. O Render fornece logs de erros e informações de desempenho.

Plano de Testes

Objetivo: Testar todos os endpoints da API para garantir que estão funcionando como esperado e que as respostas estão corretas. Nessa etapa é utilizada a aplicação Postman, configurado para a URL: <https://vitibrasil.onrender.com/login> para execução das requisições HTTP.

1. Testes dos Endpoints:

- **Login (GET)**

- **Descrição:** Verifica se a página de login é retornada corretamente.
- **Endpoint:** <https://vitibrasil.onrender.com/login>
- **Método:** GET
- **Expectativa:** Resposta 200 OK com o formulário HTML.

- **Login (POST)**

- **Descrição:** Testa a autenticação do usuário.
- **Endpoint:** <https://vitibrasil.onrender.com/login>
- **Método:** POST
- **Body (JSON):**

```
json
Copiar código
{
  "username": "user1",
  "password": "password1"
}
```

- **Expectativas:**
 - 200 OK: Retorna um token JWT.
 - 401 Unauthorized: Mensagem de erro para credenciais inválidas.
 - 400 Bad Request: Mensagem de erro se o nome de usuário ou senha estiverem ausentes.

- **Download de Arquivos (GET) (implementação futura com Banco de Dados)**

- **Descrição:** Testa o download de arquivos CSV.
- **Endpoint:** <https://vitibrasil.onrender.com/download/>
- **Método:** GET
- **Headers:**
 - Authorization: Bearer {{token}}
- **Expectativas:**
 - 200 OK: O arquivo CSV deve ser retornado como anexo.
 - 401 Unauthorized: Mensagem de erro se o token JWT for inválido ou expirado.
 - 404 Not Found: Mensagem de erro se o arquivo não for encontrado.

- **Download Dimensão País (Implementação futura)**

- **Descrição:** Verifica o download do arquivo CSV de dimensão país.
- **Endpoint:** https://vitibrasil.onrender.com/download/dimensao_pais
- **Método:** GET
- **Expectativas:**
 - 200 OK: O arquivo CSV deve ser retornado como anexo.
 - 500 Internal Server Error: Mensagem de erro em caso de falha ao enviar o arquivo.

Cenários de Uso:

1. Análise de Dados para Comércio Internacional:

- Empresas ou pesquisadores podem usar a aplicação para baixar e analisar dados sobre a produção, importação e exportação de diferentes países, ajudando a identificar padrões de tendências e oportunidades de mercado.

2. Comparação de Métricas Entre Países:

- Os gráficos gerados pela aplicação fornecem uma comparação visual clara das atividades econômicas de diferentes países, facilitando a identificação dos maiores produtores, importadores e exportadores.

3. Consultoria Econômica e Estatística:

- Consultores podem usar os dados da "Dimensão País" para oferecer insights a clientes sobre o desempenho de países em diversas categorias, além de garantir que os dados estejam consolidados e sem duplicidades.

4. Download e Processamento de Dados:

- Os arquivos CSV podem ser baixados para uso em outros sistemas de análise de dados, tornando a aplicação uma ferramenta útil para quem precisa integrar esses dados em relatórios mais amplos ou realizar análises customizadas.
- A API para baixar periodicamente dados de CSV sobre produção e comercialização do site da EMBRAPA e alimentar um modelo de Machine Learning para previsões e análises futuras

5. Relações Esperadas (Correlações positivas)

- **Produção X Processamento:** Existe uma relação direta entre a produção de uvas e a quantidade de uvas processadas para a produção de vinhos e outros derivados. Um aumento na produção, geralmente, leva a um aumento no processamento.
- **Processamento X Comercialização:** A quantidade de produtos processados está diretamente relacionada à quantidade de produtos disponíveis para comercialização. Um maior volume de produtos processados, em geral, resulta em um maior volume de produtos comercializados.
- **Comercialização X Importação X Exportação:** A comercialização pode ser influenciada pela importação e exportação. Um aumento nas exportações, por exemplo, pode reduzir a necessidade de importação e vice-versa. A competitividade dos produtos nacionais no mercado internacional também influencia a comercialização.

Usuários: O treinamento para os usuários da API é dividido em três fases principais: o Treinamento Inicial, que visa familiarizar os funcionários com o funcionamento da API, procedimentos de segurança e funcionalidades de análise de dados sobre a viticultura brasileira, através de sessões práticas, teóricas, manuais e tutoriais; a Capacitação Contínua, focada em manter os funcionários atualizados sobre novas funcionalidades e melhorias, por meio de workshops periódicos, webinars e atualizações regulares via e-mail; e a Documentação e Suporte, que oferece materiais de apoio, FAQs detalhadas e suporte técnico contínuo para solucionar dúvidas e problemas.

Melhoria Contínua: Avaliar o desempenho da API e identificar áreas para melhorias com Monitoramento de logs, feedback dos usuários e análises de desempenho.