FACULDADE IMPACTA
PROGRAMA DE PÓS-GRADUAÇÃO EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

# Projeto Flask

Beatriz Bramont Oliveira
Giovanna Petrilli Venditti
Isadora Cristyne de Lima Silva
Larissa Costa Silva
Vinicios de Lima Basteiro

Objetivo: Apresentar a eficiência do framework Flask na linguagem Python, colocando em prática conceitos como Interface de Programação de Aplicações (API 's), dicionários, programação orientada a objetos, tratamentos de erro e TDD (Test-Driven Development).

São Paulo
19/03/2025

Faculdade IMPACTA

# Relatório
# 1

# Relatório Projeto Flask - Entrega 1

## 1. Introdução

O Flask é um framework web em Python, ideal para criar APIs, com fácil integração com bancos de dados e suporte a testes automatizados. Sua simplicidade e flexibilidade fazem dele uma das ferramentas mais populares, especialmente em ambientes educacionais ou para protótipos rápidos. Ele facilita a criação de rotas e o manuseio de requisições HTTP, sendo ótimo para projetos que ensinam e demonstram conceitos técnicos de forma prática.

## 2. Descrição e Análise do Caso

Neste projeto, foi analisado o benefício de usar o Flask para obter dados correlacionados, aplicando-o em um sistema real de gestão educacional. Para o desenvolvimento, utilizamos o Visual Studio Code como ambiente de codificação, o Postman para testar as requisições GET e POST, e o Git com o GitHub para controle de versão e compartilhamento do código. O repositório foi dividido em diferentes branches para separar os códigos em desenvolvimento dos já concluídos, o que ajudou a garantir que os programas funcionando bem não se perdessem durante as edições ou commits.

## 3. Implementação ou Procedimento

O objetivo principal do projeto é criar um sistema para gerenciar informações sobre uma instituição educacional, permitindo criar, ler, atualizar e (potencialmente) excluir dados de alunos, professores e turmas. Os dados são armazenados em um dicionário, que serve como um banco de dados em memória para demonstração.

**Endpoints da API:**

- **Criação (POST):**
  - **/alunos (POST):** Adiciona um aluno ao dicionário de alunos, verificando se o turma_id existe.
  - **/professor (POST):** Adiciona um professor ao dicionário de professores.
  - **/turma (POST):** Adiciona uma turma ao dicionário de turmas, verificando se o professor_id existe.
- **Leitura (GET):**
  - **/alunos (GET):** Retorna todos os alunos como JSON.
  - **/professor (GET):** Retorna todos os professores como JSON.
  - **/turma (GET):** Retorna todas as turmas como JSON.
- **Atualização (PUT):**
  - **/alunos/int:idAluno (PUT):** Atualiza um aluno específico.
  - **/professor/int:idProfessor (PUT):** Atualiza um professor específico.
  - **/turma/int:idTurma (PUT):** Atualiza uma turma específica.
- **Deletar (DELETE):**
  - **/alunos/int:idAluno (DELETE):** Deleta um aluno.
  - **/professor/int:idProfessor (DELETE):** Deleta um professor.
  - **/turma/int:idTurma (DELETE):** Deleta uma turma.

**Funções utilizadas no Flask:**

- **Flask(name):** Cria a instância do aplicativo Flask.
- **@app.route():** Associa URLs a funções Python.
- **request.json:** Acessa dados JSON nas requisições POST e PUT.

- **jsonify():** Converte dados Python para JSON.
- **app.run(debug=True):** Inicia o servidor Flask em modo de depuração.

**Desafios:** Um dos principais desafios foi a criação de testes unitários devido à complexidade do sistema. Para a organização do projeto, as tarefas foram distribuídas entre os membros da equipe, com problemas de comunicação resolvidos via WhatsApp.

**Divisão de tarefas:**

- Líder: Beatriz Bramont Oliveira
    - GitHub, controle de versão, implementação das rotas CRUD, tratamento de erros, criação de TDD, testes no Postman.
- Isadora: Controle de versão, testes com Postman, TDD.
- Larissa: Implementação do DELETE.
- Todos: Colaboração no relatório e testes.

## 4. Resultados

Este projeto foi estruturado para receber informações do usuário e evitar erros no envio de JSON. O sistema possui validações para garantir que os dados sejam válidos antes de serem armazenados, como a verificação de IDs duplicados e a validação de campos nulos. Além disso, as entidades são interligadas por IDs, e registros só são válidos se as entidades relacionadas existirem, como garantir que um aluno tenha um idTurma válido e uma turma tenha um idProfessor válido.

Os testes realizados tinham o objetivo de verificar as operações CRUD (criação, leitura, atualização e exclusão) da API para professores, turmas e alunos. Utilizou-se a biblioteca unittest do Python e a biblioteca requests para realizar chamadas HTTP aos endpoints. Os testes incluíram:

- **Criação (POST):** Verificar se a criação de professores, turmas e alunos com dados válidos retornava o código de status correto.
- **Atualização (PUT):** Testar a atualização de dados de professores, turmas e alunos e verificar o código de status.

- **Exclusão (DELETE):** Testar a exclusão de professores, turmas e alunos e validar o código de status.
- **Listagem (GET):** Verificar se os recursos criados eram listados corretamente.

### Os resultados atenderam às expectativas?

Sim, os resultados atenderam às expectativas. Os testes mostraram que as operações básicas do CRUD (Create, Read, Update, Delete) funcionaram como esperado. A maioria dos testes confirmou que as ações de criação, atualização e exclusão retornaram os códigos de status HTTP corretos, e os dados criados foram armazenados e puderam ser lidos posteriormente. Em resumo, as funcionalidades do CRUD da API foram atendidas com sucesso.

### O que eles revelam sobre o problema ou caso analisado?

Os resultados indicam que a API para gerenciar professores, turmas e alunos está funcionando corretamente. No entanto, os testes realizados são apenas básicos e de unidade, e uma avaliação mais completa do sistema requer testes mais abrangentes, como:

- **Testes de casos de erro**: Verificar o comportamento da API com entradas inválidas ou dados ausentes.
- **Testes de integração**: Avaliar como diferentes partes da API interagem entre si e com outros sistemas.
- **Testes de desempenho**: Testar a capacidade de resposta da API sob diversas cargas de trabalho.
- **Testes de segurança**: Garantir que a API seja protegida contra vulnerabilidades.

Apesar disso, os testes realizados mostram que as funcionalidades básicas de gerenciamento de dados estão implementadas e funcionando corretamente conforme esperado.

## 5. Conclusão

O código serve como um protótipo útil para compreender os conceitos básicos de criação de uma API REST com Flask. No entanto, para ser usado em um ambiente de maior carga de dados, será necessário realizar otimizações significativas em termos de escalabilidade, desempenho e robustez. A substituição

do armazenamento em memória por um banco de dados real, o uso de um servidor adequado e a implementação de cache são passos essenciais para transformar este protótipo em um sistema capaz de lidar com grandes volumes de dados e requisições simultâneas.

## 6. Impacto e Conexão com o Mundo Real

A API desenvolvida simula um sistema de gerenciamento de dados, usando princípios RESTful e métodos HTTP para operações CRUD, comuns em aplicações web e móveis. A inclusão de testes unitários destaca a importância da qualidade e confiabilidade das APIs.

Fora do ambiente acadêmico, o aprendizado de criação e teste de APIs é essencial para desenvolvedores, permitindo integrar sistemas, fornecer dados para aplicativos móveis e criar microsserviços. APIs também otimizam fluxos de trabalho, melhoram a troca de informações entre sistemas e têm aplicação em áreas como autenticação, monitoramento em tempo real e sistemas de pagamento.

## 7. Desafios Futuros e Melhorias

Entrando em consenso, o grupo acredita que a melhoria na gestão de tarefas é um ponto importante a ser levado em consideração nas próximas etapas do projeto, tendo em vista a suma importância de uma boa organização na divisão de responsabilidades. Outro ponto importante é a otimização de tempo, o planejamento prévio para a elaboração das etapas do projeto é importante para não deixar que fiquem muitas alterações e implementações dentro da API a serem realizadas em um curto período de tempo.

Melhorias futuras também serão implementadas ao projeto, como por exemplo a integração a um banco de dados, visto que teremos um número alto de informações a serem armazenadas e executadas.

Faculdade IMPACTA

## Referências


API de Gestão Escolar. Disponível em: <https://school-system-spi.onrender.com/docs>.

DAIANA S. Flask Python: como usar e qual sua função? | Homehost. Disponível em: <https://www.homehost.com.br/blog/pythondjango/flask-python/#Aplicacao_com_3_rotas_e_padrao_REST_GET_POST_PUT_DELETE>.

“Test Coverage — Flask Documentation (3.1.x).” Palletsprojects.com, 2018, flask.palletsprojects.com/en/stable/tutorial/tests/.

POKEMAOBR, R. C. Flask Python: o que é e como funciona? Disponível em: <https://www.locaweb.com.br/blog/temas/codigo-aberto/flask-phyton-o-que-e/>.

MONTEIRO, V. N.; CARVALHO, G. A.; CARDOSO, V. Integração de tecnologias: python, arduino, redes neurais e flask para soluções tecnológicas eficientes. Apoena, v. 6, p. 224–229, 2023.

Nataniel Paiva – Medium. Disponível em: <https://nataniel-paiva.medium.com/>.

Arquiteto Cloud. Disponível em: <https://arquitetocloud.com/>.

HTTPS://PLUS.GOOGLE.COM/U/0/+DATACAMP. Learn R, Python & Data Science Online. Disponível em: <https://www.datacamp.com/>.