

## Desenvolvimento de um Sistema Web em Python com Flask

### Objetivo

Desenvolver um **sistema web dinâmico** utilizando **Python com Flask**, aplicando os principais conceitos de **programação orientada a objetos**, **estruturação de rotas**, **templates HTML**, **CSS** e **Bootstrap**.

### Requisitos do sistema

#### 1. Estrutura do projeto

O projeto deve seguir a **estrutura básica do Flask**, contendo:

- ✓ app.py (arquivo principal) Exemplo: cadastrar, listar, editar, excluir, consultar, sair.
- ✓ Pasta templates/ para os arquivos HTML
- ✓ Pasta static/ para CSS, JS e imagens

#### 2. O sistema deve possuir rotas para as principais operações (CRUD)

- ✓ / → Página inicial
- ✓ /cadastrar → Cadastro de registros
- ✓ listar → Listagem de registros
- ✓ /editar/<id> → Edição

#### 3. Interface WEB

- ✓ As páginas devem ser criadas com **HTML** e **CSS**. Pode ser usado Bootstrap
- ✓ O sistema deve ter um **menu de navegação** com links para as principais rotas.
- ✓ As páginas devem manter um **layout organizado**, usando o conceito de **template base** do Flask (ex.: base.html com {% block content %})

#### 4. Exceções

- ✓ O sistema deve tratar erros com try/except, como:
  - Consultar um item que não existe.
  - Inserir valores inválidos.

## 5. Classes e objetos

- ✓ Crie uma classe principal (ex.: Produto, Aluno, Paciente, Veículo), com atributos privados.
- ✓ Use @property e @setter para controlar pelo menos um atributo (ex.: preço, nota, saldo, idade).

## 6. Funções e Rotas

- ✓ Cada funcionalidade (cadastrar, listar, editar, excluir) deve ter **uma função e uma rota específica** no Flask.
- ✓ Deve haver **validação de dados** (ex.: campos obrigatórios, valores inválidos).
- ✓ Utilize **métodos GET e POST** adequadamente:

GET → exibir formulários e páginas

POST → receber e processar dados enviados pelos formulários

## 7. Conexão com banco de dados – Para equipe com 4 estudantes ou mais.

- ✓ O SGDB fica a sua escolha.

---

### Critério avaliativo:

- Funcionamento geral do sistema **CRUD + ROTAS** **2.5**

O sistema deve realizar corretamente as operações de cadastro, listagem, edição e exclusão.

- Estrutura do código **2.5**

Código bem estrutura com uso do flask com rotas

- Interface (HTML, CSS, Bootstrap) **1.0**

Layout organizado, interface adequada

- Apresentação e explicação do projeto **4,0**