Mini-Sistema de Biblioteca – Registro de Livros

Isabele Fernanda da Silva Albano; Emanoelly Francinny Brito Tavares; Lívia Tainá de Medeiros Oliveira; Tamiris dos Santos Medeiros

Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte – Campus Caicó

Técnico de Nível Médio em Informática para Internet – Disciplina PSI

Professor: Romerito Campos

isabele.albano@escolar.ifrn.edu.br,Francinny.brito@escolar.ifrn.edu.br,livia.o@escolar.ifrn.edu.br,tamiris.medeiros@escolar.ifrn.edu.br

Resumo. *Este artigo apresenta o desenvolvimento de um mini-sistema de biblioteca, cujo objetivo é permitir o cadastro e o gerenciamento de livros de forma simples e organizada. O projeto foi desenvolvido como atividade da disciplina de Programação para Sistemas de Informação (PSI) e segue um modelo de API REST com uma única entidade principal. A aplicação possibilita criar, listar, editar, filtrar, atualizar o status e remover registros de livros, utilizando persistência em arquivo JSON e uma interface web básica para interação do usuário.*

Palavras-chave: biblioteca, API REST, sistemas web, CRUD, Flask.

1. Introdução

Com a crescente digitalização de processos, sistemas simples de registro tornam-se fundamentais para a organização de informações em diferentes contextos. No ambiente educacional, o desenvolvimento de aplicações desse tipo contribui para a consolidação dos conceitos de backend, frontend e integração entre sistemas.

Este trabalho tem como objetivo apresentar a construção de um mini-sistema de biblioteca, cujo foco é o cadastro e o gerenciamento de livros. O sistema foi desenvolvido respeitando as regras propostas pelo professor da disciplina de PSI,

utilizando apenas uma entidade principal e sem autenticação de usuários ou relacionamentos entre tabelas.

2. Metodologia

O desenvolvimento do projeto foi dividido em três partes principais: definição do modelo de dados, implementação da API e construção do frontend.

2.1 Stack utilizada

- **Backend:** Python com Flask
- **Persistência:** Arquivo JSON (items.json)
- **Frontend:** HTML, CSS e JavaScript
- **Comunicação:** API REST utilizando requisições HTTP (GET, POST, PUT, PATCH e DELETE)

2.2 Modelo de dados

O sistema possui apenas uma entidade principal chamada **Item**, representando um livro. Os campos utilizados foram:

- id (number): gerado automaticamente pelo backend
- titulo (string): obrigatório, mínimo de 3 caracteres
- tipo (string): definido como "livro"
- status (string): valores permitidos ["ativo", "concluído", "arquivado"]
- descricao (string): campo opcional utilizado para informar o autor
- data (string): data de cadastro no formato YYYY-MM-DD

2.3 Endpoints da API

A API foi desenvolvida seguindo o padrão REST, com base URL http://localhost:5000.

- **GET /items:** lista todos os livros cadastrados
- **GET /items?tipo=livro:** filtra livros pelo tipo
- **GET /items?status=ativo:** filtra livros pelo status
- **POST /items:** cria um novo livro

- **PUT /items/:id:** edita todas as informações de um livro
- **PATCH /items/:id/status:** altera apenas o status do livro
- **DELETE /items/:id:** remove um livro

Todas as rotas realizam validações conforme as regras definidas no projeto.

3. Resultados

Como resultado, obteve-se uma aplicação funcional que permite o gerenciamento completo dos livros cadastrados. A interface web apresenta um formulário para criação e edição dos registros, uma lista com todos os livros e opções para filtrar por tipo e status.

A API retorna respostas em formato JSON, tanto em casos de sucesso quanto de erro de validação.

Exemplo de JSON retornado


```
{
  "id": 1,
  "titulo": "Dom Casmurro",
  "tipo": "livro",
  "status": "ativo",
  "descricao": "Machado de Assis",
  "data": "2026-01-25"
}
```


4. Conclusão

O mini-sistema de biblioteca atendeu aos objetivos propostos, permitindo o cadastro e a gestão de livros de forma simples e eficiente. O projeto contribuiu para a prática dos conceitos de API REST, validação de dados, persistência em arquivo JSON e integração entre frontend e backend.

Como limitações, destaca-se a ausência de autenticação de usuários e a utilização de apenas uma entidade. Como melhorias futuras, sugere-se a inclusão de novos campos, paginação de resultados e, em um cenário mais avançado, a adoção de um banco de dados relacional.

Referências

FLASK. *Flask Documentation*. Disponível em: https://flask.palletsprojects.com/.

FIELDING, R. T. *Architectural Styles and the Design of Network-based Software Architectures*. 2000.

MDN Web Docs. *HTTP Methods*. Disponível em: https://developer.mozilla.org/.