UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE – UERN
Pró-Reitoria de Ensino e Graduação - PROEG
Unidade Universitária: Campus de Natal – CaN
Ciência da Computação

📝 Documento de Planejamento – AT1

1. Nome e descrição do projeto

Nome: Gerenciador de Mangás
Descrição:
Este projeto é uma aplicação web que simula o gerenciamento de mangás em uma livraria. A aplicação foi desenvolvida utilizando o framework Django para o back-end e HTML + JavaScript puro no front-end.

O projeto implementa uma API RESTful com as operações básicas de uma entidade, **Manga** permitindo a visualização de todos os registros, visualização individual por ID, criação e exclusão. Os dados são armazenados apenas em memória (não persistem após reiniciar o servidor), como definido para esta etapa da atividade.

---

2. Tema e entidade escolhida

Tema: Catálogo de produtos de uma livraria.
Entidade: Manga — representa um mangá com seus atributos principais, como título, autor, quantidade de volumes, status de disponibilidade e preço.

---

3. Estrutura dos dados (JSON)

A estrutura de cada objeto da entidade `Manga` é representada no formato JSON da seguinte forma:

```
{

  "id": 1,
  "titulo": "One Piece",
  "autor": "Eiichiro Oda",
  "volumes": 105,
  "status": "Disponível",
  "preco": 39.90
}
```

Explicação dos campos:

- `id`: identificador numérico gerado automaticamente.
- `titulo`: nome do mangá.
- `autor`: criador da obra.
- `volumes`: número total de volumes disponíveis.
- `status`: texto indicando se o mangá está disponível.
- `preco`: valor monetário do mangá.

Os dados são armazenados em uma lista Python simulando um banco em memória:

```
mangas = []
next_id = 1
```

---

4. Lista de rotas REST utilizadas

A aplicação implementa as seguintes rotas REST no back-end, utilizando funções definidas diretamente no `views.py`. A aplicação implementa as seguintes rotas REST no back-end:

Método HTTP	Rota	Função
GET	`/mangas/`	Retorna todos os mangás cadastrados.
GET	`/mangas/<id>/`	Retorna os dados de um mangá específico.
POST	`/mangas/`	Cria um novo mangá a partir do JSON enviado.
DELETE	`/mangas/<id>/`	Remove um mangá específico.

Como funciona cada método:

- **GET /mangas/**
  Retorna uma lista completa de todos os mangás armazenados na lista `mangas`. Caso esteja vazia, retorna uma mensagem informando que nenhum mangá foi cadastrado ainda.
  O código verifica isso com:

```
if not mangas:
    return JsonResponse({"mensagem": "Nenhum mangá
cadastrado."})
```

- **GET /mangas/<id>/**
  Busca e retorna um mangá com o ID fornecido na URL. Se não encontrar, retorna erro 404:

```
manga = next((m for m in mangas if m["id"] == id), None)
if not manga:
    return JsonResponse({'erro': 'Mangá não encontrado'},
status=404)
```

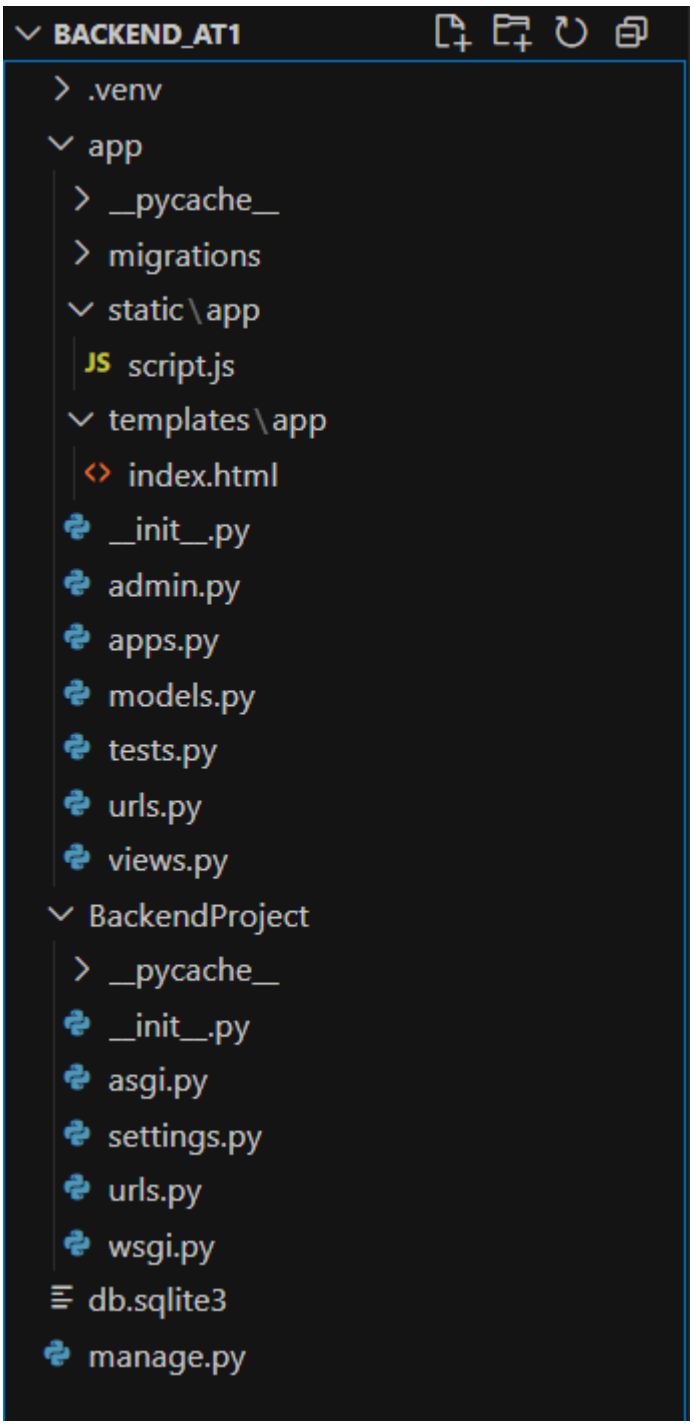
- **POST /mangas/**
  Cria um novo registro com os dados enviados em formato JSON. O ID é gerado automaticamente por meio da variável `next_id`, que é incrementada manualmente.

```
novo = {
    "id": next_id,
    ...
}
mangas.append(novo)
next_id += 1
```

- **DELETE /mangas/<id>/**
  Remove um mangá com base no ID. A lista é reatribuída excluindo o item:

```
mangas = [m for m in mangas if m["id"] != id]
```

5. Estrutura geral da aplicação



Roteamento e exibição de páginas:

- A **página inicial (`/`)** exibe uma mensagem de boas-vindas usando `HttpResponse`.
- A **página `/mangas/interface/`** carrega o HTML e JavaScript que consome a API via `fetch()`.

```
def home(request):
```

```
    return HttpResponse(
        "<h1>Bem-vindos ao gerenciador de mangás!</h1>"
        "<p>Vá para <a
href='/mangas/interface/'>/mangas/interface/</a> "
        "para acessar a interface.</p>"
    )

def pagina_mangas(request):
    return render(request, 'app/index.html')
```

Funcionalidades da interface (`index.html`):

- Botão para listar os mangás (com toggle mostrar/esconder).
- Formulário para adicionar um novo mangá.
- Campo para buscar um mangá por ID.
- Botão de deletar ao lado de cada item listado.

O projeto foi desenvolvido utilizando Django, porém **sem banco de dados ativo** nesta etapa da atividade. Por isso, os dados não persistem ao reiniciar o servidor — isso foi proposital, conforme exigência da descrição da atividade.

✳ **Por que os dados não são salvos?**

Durante a execução, os mangás são armazenados apenas em memória, na variável `mangas = []`. Como isso não está ligado a nenhuma base de dados (como SQLite ou PostgreSQL), **toda informação é perdida quando o servidor é encerrado**.

Essa abordagem é útil para simulações rápidas ou ambientes de teste, e será substituída na próxima etapa da atividade, quando o modelo `Manga` será integrado ao banco de dados com uso do `models.py` e `ORM do Django`.