Desenvolvimento de API - Farmácia Já

Checkpoint 02

Definição do escopo:

A ideia é que o usuário possa pesquisar o nome de um medicamento e descobrir quais farmácias o possuem, além de ter a opção de filtrar os resultados por bairro para facilitar encontrar o local mais próximo.

- Criar uma API para consulta de farmácias e medicamentos.
- Permitir pesquisa por nome do medicamento e filtro por bairro.
- Cadastrar, editar e remover farmácias e medicamentos.
- Estabelecer um relacionamento **muitos-para-muitos** entre farmácias e medicamentos.
- Retornar as informações em formato **JSON**.
- Armazenar os dados em um **banco PostgreSQL**.
- Criar uma **documentação dos endpoints** da API.

Definição dos endpoints:

Usuários

GET /usuarios

- **Função:** Retorna a lista de todos os usuários cadastrados.
- **Busca:** sem parâmetros.

```
GET /usuarios
Host: api.farmaciaja.com
Accept: application/json
```

GET /usuarios/:id

- **Função:** Retorna os dados de um usuário específico.
- **Busca:** `params` → `id`

```
GET /usuarios/5
Host: api.farmaciaja.com
Accept: application/json
```

Resposta:

```
{
 "id": 5,
 "nome": "Flávio Santos",
 "telefone": "(11) 98888-2222",
 "email": "flavio@email.com",
 "cpf": "123.456.789-00",
 "endereco": "Rua das Flores, 50 - Centro"
}
```

POST /usuarios

- **Função:** Cadastra um novo usuário.
- **Busca:** body
- **Headers:** `Content-Type: application/json`

```
POST /usuarios
Host: api.farmaciaja.com
Content-Type: application/json

{
 "nome": "Flávio Santos",
 "telefone": "(11) 98888-2222",
 "email": "flavio@email.com",
 "cpf": "123.456.789-00",
 "endereco": "Rua das Flores, 50 - Centro"
}
```

PUT /usuarios/:id

- **Função:** Atualiza os dados de um usuário existente.
- **Busca:** `params` → id; body → novos dados.

```
PUT /usuarios/5
Host: api.farmaciaja.com
Content-Type: application/json

{
 "telefone": "(11) 97777-3333",
 "endereco": "Av. Paulista, 1000"
}
```

DELETE /usuarios/:id

- **Função:** Remove um usuário.
- **Busca:** `params` → id.

```
DELETE /usuarios/5
Host: api.farmaciaja.com
```

Farmácias

GET /farmacias

- Retorna todas as farmácias.
- **Busca:** sem parâmetros.

```
GET /farmacias
Host: api.farmaciaja.com
Accept: application/json
```

GET /farmacias/:id

- Retorna detalhes de uma farmácia específica.
- **Busca:** params → id

```
GET /farmacias/12
Host: api.farmaciaja.com
```

POST /farmacias

- Cadastra uma nova farmácia.
- **Busca:** body

```
POST /farmacias
Host: api.farmaciaja.com
Content-Type: application/json

{
 "nome": "Farmácia São José",
 "endereco": "Rua das Flores, 120",
 "bairro": "Centro",
 "telefone": "(11) 98877-6655",
 "entregas": true
}
```

PUT /farmacias/:id

- Atualiza informações de uma farmácia.
- **Busca:** params → id; body → novos dados.

```
PUT /farmacias/12
```

```
Host: api.farmaciaja.com
Content-Type: application/json

{
 "telefone": "(11) 97777-1111",
 "entregas": false
}
```

DELETE /farmacias/:id

- Remove uma farmácia.
- **Busca:** params → id.

```
DELETE /farmacias/12
Host: api.farmaciaja.com
```

Medicamentos

GET /medicamentos

- Retorna todos os medicamentos cadastrados.
- **Busca:** sem parâmetros.

```
GET /medicamentos
Host: api.farmaciaja.com
Accept: application/json
```

GET /medicamentos/:id

- Retorna informações detalhadas de um medicamento.
- **Busca:** params → id

```
GET /medicamentos/5
Host: api.farmaciaja.com
```

POST /medicamentos

- Cadastra um novo medicamento.
- **Busca:** body

```
POST /medicamentos
Host: api.farmaciaja.com
Content-Type: application/json
```

```
{
 "nome": "Dipirona Sódica 500mg",
 "fabricante": "EMS"
}
```

PUT /medicamentos/:id

- Atualiza informações de um medicamento.
- **Busca:** params → id; body → novos dados.

```
PUT /medicamentos/5
Host: api.farmaciaja.com
Content-Type: application/json

{
 "fabricante": "Neo Química"
}
```

DELETE /medicamentos/:id

- Remove um medicamento.
- **Busca:** params → id

```
DELETE /medicamentos/5
Host: api.farmaciaja.com
```

Relação Farmácias ↔ Medicamentos

GET /farmacias/medicamentos

- **Função:** Busca farmácias que possuem medicamentos.
- **Busca:** query → nome (um ou mais)

```
GET /farmacias/medicamentos?nome=dipirona&nome=paracetamol HTTP/1.1
Host: api.farmaciaja.com
Accept: application/json
```

GET /farmacias/medicamentos?nome=&bairro=

- **Função:** Filtra farmácias por medicamento e bairro.
- **Busca:** query → nome, bairro

```
GET /farmacias/medicamentos?nome=dipirona&bairro=Centro HTTP/1.1
Host: api.farmaciaja.com
```

```
Accept: application/json
```

POST /farmacias/:id/medicamentos

- **Função:** Associa medicamentos a uma farmácia.
- **Busca:** params → id da farmácia; body → ids de medicamentos.

```
POST /farmacias/12/medicamentos
Host: api.farmaciaja.com
Content-Type: application/json

{
 "medicamentos": [1, 3, 7]
}
```

Padronização de Respostas

Sucesso

```
{
 "mensagem": "Cadastro realizado com sucesso",
 "status": 201
}
```

Erro

```
{
 "erro": "Farmácia não encontrada",
 "status": 404
}
```

OBS: Muitos endpoints poderão contar com a implementação de filtros para auxiliar o usuário a encontrar o medicamento procurado.

Levantamento de requisitos:

Funcionais:

- **RF01:** Cadastrar farmácias com informações como nome, endereço, bairro, telefone e se realiza entregas.
- **RF02:** Cadastrar medicamentos disponíveis no programa Farmácia Popular.
- **RF03:** Associar medicamentos às farmácias que os possuem.
- **RF04:** Permitir a busca de farmácias que tenham determinado medicamento.
- **RF05:** Permitir filtragem por bairro para facilitar a busca.
- **RF06:** Atualizar e remover farmácias e medicamentos quando necessário.

Não funcionais:

- **RNF01:** O sistema deve ser desenvolvido em **Node.js com TypeScript**.
- **RNF02:** O banco de dados deve ser **PostgreSQL**.
- **RNF03:** A API deve seguir o padrão **RESTful**, retornando os dados em **JSON**.
- **RNF04:** O código deve ser organizado em **estrutura MVC** (controllers, services, routes, modells).
- **RNF05:** Deve haver **tratamento de erros** e mensagens claras de retorno.
- **RNF06:** Informações sensíveis devem ficar em um arquivo **.env**.
- **RNF07:** A API deve possuir **documentação dos endpoints** (Postman).