### Funcionamento da API REST em Java utilizando Spring Boot

#### Estrutura e Funcionamento

1. **Controlador REST (EmployeeController)**:
   * Utiliza anotações Spring Boot para mapear requisições HTTP para métodos de manipulação de dados.
   * Gerencia requisições POST para adicionar empregados e GET para recuperar empregados.
2. **Modelo de Dados (Employee)**:
   * Representa os dados de um empregado.
   * Utiliza encapsulamento para proteger os dados dos empregados.
3. **Armazenamento em Memória**:
   * Utiliza uma lista (List<Employee>) para armazenar os objetos Employee em memória.
   * Permite adicionar múltiplos empregados de uma vez.

#### Conceitos de POO Utilizados

1. **Encapsulamento**:
   * A classe Employee tem atributos privados e métodos públicos get e set para acessar e modificar esses atributos.
2. **Abstração**:
   * A API REST abstrai os detalhes de como os dados dos empregados são armazenados e manipulados, proporcionando uma interface simples para o cliente.
3. **Composição**:
   * O controlador (EmployeeController) usa a classe Employee como parte de sua composição para gerenciar os dados dos empregados.

#### Estrutura de Dados Utilizada

* **Lista (List<Employee>)**:
  + Utilizada para armazenar e gerenciar os objetos Employee.
  + Permite a adição e recuperação eficiente dos dados dos empregados.

**Fluxograma do Sistema**

plaintext

Copiar código

┌────────────────────────────────────────────┐

│ Início │

└─────────────┬──────────────────────────────┘

│

▼

┌────────────────────────────────────────────┐

│ Receber Requisição POST /employees │

│ - Corpo: Lista de Empregados (JSON) │

└─────────────┬──────────────────────────────┘

│

▼

┌────────────────────────────────────────────┐

│ Adicionar Empregados na Lista (Memória) │

└─────────────┬──────────────────────────────┘

│

▼

┌────────────────────────────────────────────┐

│ Retornar Empregados Adicionados │

└─────────────┬──────────────────────────────┘

│

▼

┌────────────────────────────────────────────┐

│ Receber Requisição GET /employees/{id} │

│ - Parâmetro: ID do Empregado │

└─────────────┬──────────────────────────────┘

│

▼

┌────────────────────────────────────────────┐

│ Procurar Empregado na Lista (Memória) │

│ - Retornar Empregado Correspondente │

│ ou null se não encontrado │

└─────────────┬──────────────────────────────┘

│

▼

┌────────────────────────────────────────────┐

│ Receber Requisição GET /employees │

│ - Sem Parâmetros │

└─────────────┬──────────────────────────────┘

│

▼

┌────────────────────────────────────────────┐

│ Retornar Lista Completa de Empregados │

└─────────────┬──────────────────────────────┘

│

▼

┌────────────────────────────────────────────┐

│ Fim │

└────────────────────────────────────────────┘

**Explicação Detalhada**

1. **Controlador REST**:
   * Anotado com @RestController para indicar que é um controlador REST.
   * Mapeia as requisições para os métodos com @RequestMapping, @PostMapping e @GetMapping.
2. **Método POST**:
   * Mapeado para /employees para adicionar múltiplos empregados.
   * Recebe uma lista de empregados no corpo da requisição e adiciona à lista em memória (employeeRepository).
3. **Método GET por ID**:
   * Mapeado para /employees/{id} para recuperar um empregado específico.
   * Utiliza stream para buscar na lista pelo ID do empregado.
4. **Método GET Todos os Empregados**:
   * Mapeado para /employees para recuperar todos os empregados.
   * Retorna a lista completa de empregados armazenados.