Documentação do Projeto Angular e Java + MySQL

1. Introdução

Este documento fornece uma visão geral do projeto desenvolvido utilizando Angular para o frontend, Java (com Spring Boot) para o backend e MySQL como banco de dados relacional. O sistema implementa funcionalidades básicas de gerenciamento de pautas e votações.

2. Arquitetura Geral

O projeto é dividido em duas partes principais:

- **Frontend (Angular)**: Responsável pela interface de usuário, interação com o usuário final e consumo de APIs do backend.
- **Backend (Java + Spring Boot)**: Fornecimento de APIs RESTful para acesso aos dados, lógica de negócios e integração com o banco de dados MySQL.

3. Tecnologias Utilizadas

- **Angular**: Framework de desenvolvimento front-end baseado em TypeScript. Inclui RxJS para manipulação de operações assíncronas.
- **Java**: Linguagem de programação para desenvolvimento do backend.
- **Spring Boot**: Framework Java para criação de APIs RESTful e aplicações web.
- **MySQL**: Sistema de gerenciamento de banco de dados relacional.

4. Funcionalidades Implementadas

Frontend (Angular)

- **Login e Cadastro de Usuários**: Gerenciamento de autenticação de votantes.
- **Gerenciamento de Pautas**: CRUD básico para pautas, incluindo iniciar e encerrar votações.
- **Contador Regressivo**: Exibição de um contador regressivo para as votações iniciadas.

Backend (Java + Spring Boot)

- **API RESTful**: Fornecimento de endpoints para CRUD de pautas e operações de votação.
- **Segurança**: Controle de acesso baseado em tokens JWT para autenticação de usuários.
- **Persistência de Dados**: Integração com o MySQL para armazenamento de pautas e usuários.

5. Estrutura de Diretórios

- **frontend/**: Código fonte do frontend em Angular.
- **src/main/java/**: Código fonte do backend em Java.
    - **config/**: Configurações adicionais (por exemplo, CORS, segurança).
    - **controller/**: Controladores REST para gerenciar requisições HTTP.
    - **model/**: Entidades e modelos de dados.
    - **repository/**: Interfaces de repositório para interação com o banco de dados.
    - **service/**: Lógica de negócios e serviços.

6. Instalação e Configuração

Pré-requisitos

- Node.js
- Angular CLI
- JDK
- MySQL Server

Passos de Instalação

1. **Frontend (Angular)**:
    - Clone o repositório.
    - Instale as dependências: `npm install`.
    - Inicie o servidor de desenvolvimento: `ng serve`.
2. **Backend (Java + Spring Boot)**:
    - Clone o repositório.
    - Configure o arquivo `application.properties` com as credenciais do banco de dados MySQL.
    - Execute a aplicação através do IDE ou linha de comando.

7. Uso

- Acesse a aplicação através do navegador em `http://localhost:4200`.
- Realize o login com um usuário válido ou cadastre um novo usuário.
- Gerencie pautas: crie, liste, inicie votações e encerre votações.
- Observe o contador regressivo ao iniciar uma votação.

8. Exemplos de Código

Exemplo de Controller (Java)

java

Copiar código

- `@RestController`
- `@RequestMapping("/api/pautas")`
- `@CrossOrigin(origins = "http://localhost:4200")`
- `public class PautaController {`

```
    @Autowired
    private PautaService pautaService;

    @GetMapping("/{id}")
    public ResponseEntity<Pauta> getPautaById(@PathVariable
Long id) {
        Pauta pauta = pautaService.getPautaById(id);
        if (pauta != null) {
            return ResponseEntity.ok(pauta);
        } else {
            return ResponseEntity.notFound().build();
        }
    }

    // Outros métodos do controlador
}
```

Exemplo de Componente (Angular)

typescript

Copiar código

```
import { Component, OnInit } from '@angular/core';
import { BackendService } from '../backend.service';

@Component({
  selector: 'app-gerenciamento-pautas',
  templateUrl: './gerenciamento-pautas.component.html',
  styleUrls: ['./gerenciamento-pautas.component.css']
})
export class GerenciamentoPautasComponent implements OnInit {

  pautas: any[] = [];
  newPauta: string = '';
  mensagemVotacao: string = '';
  contador: number = 0;
  interval: any;

  constructor(private backendService: BackendService) { }
```

```
  ngOnInit() {
    this.buscarPautas();
  }

  buscarPautas() {
    this.backendService.getPautas().subscribe(pautas => {
      this.pautas = pautas;
    });
  }

  cadastrarPauta() {
this.backendService.cadastrarPauta(this.newPauta).subscribe(pauta => {
      this.pautas.push(pauta);
      this.newPauta = '';
    });
  }

  iniciarVotacao(id: number) {
    this.backendService.iniciarVotacao(id).subscribe(() => {
      this.mensagemVotacao = 'Votação Iniciada';
      this.contador = 60;
      this.startCountdown(id);
    });
  }

  encerrarVotacao(id: number) {
    clearInterval(this.interval);
    this.backendService.encerrarVotacao(id).subscribe(() => {
      this.mensagemVotacao = 'Votação Encerrada';
      this.contador = 0;
      this.buscarPautas();
    });
  }

  startCountdown(id: number) {
    this.interval = setInterval(() => {
      if (this.contador > 0) {
        this.contador--;
```

```
        } else {
          clearInterval(this.interval);
          this.encerrarVotacao(id);
        }
      }, 1000);
    }
  }
```

9. Contribuições

- Contribuições são bem-vindas. Para grandes mudanças, por favor, abra uma issue primeiro para discutir o que você gostaria de mudar.