Tutorial Inserindo Chaves

Projeto: Controle de empréstimo de chaves

Tecnologias: Spring Boot Rest e Svelte

Alunos:
Gabriela Silva Rodrigues;
Gustavo Machado Pontes;
Mateus Alves Silva;
Nathan Rodrigues dos Santos.

Adição dos métodos de inserção na API

- Acesse a classe com o nome de ChaveController com o seguinte conteúdo:

```
package com.GGMN.backend.controller;

import java.util.ArrayList;
import java.util.List;

import org.springframework.beans.BeanUtils;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.HttpStatusCode;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Controller;
import org.springframework.stereotype.Service;
import org.springframework.validation.annotation.Validated;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.GGMN.backend.dto.ChaveRecordDto;
import com.GGMN.backend.model.Chave;
import com.GGMN.backend.repository.ChaveRepository;

@RestController
@RequestMapping("/chaves")
@CrossOrigin(origins = "http://localhost:8080")
public class ChaveController {

    @Autowired
```

```
    private ChaveRepository chaveRepository;

    @PostMapping
    public ResponseEntity<Chave> saveChave(@RequestBody ChaveRecordDto
chaveRecordDto) {

        Chave chave = new Chave();
        BeanUtils.copyProperties(chaveRecordDto, chave);


        return
ResponseEntity.status(HttpStatus.CREATED).body(chaveRepository.save(chave));
    }

    @GetMapping
    public ResponseEntity<List<Chave>> getAllChaves() {

        return
ResponseEntity.status(HttpStatus.OK).body(chaveRepository.findAll());
    }

    @GetMapping("/status/{status}")
    public ResponseEntity<List<Chave>> getPerStauts(@PathVariable(value="status")
boolean status) {


        return
ResponseEntity.status(HttpStatus.OK).body(chaveRepository.findByStatus(status));
    }

    @GetMapping("/situacao/{situacao}")
    public ResponseEntity<List<Chave>>
getPerSituacao(@PathVariable(value="situacao") String situacao) {

        List<Chave> chaves = new ArrayList<>();

        for (Chave chave : chaveRepository.findBySituacao(situacao)) {
            if(chave.getStatus() == true) {
                chaves.add(chave);
            }
        }



        return ResponseEntity.status(HttpStatus.OK).body(chaves);
    }
}
```

iniciando o servidor do SpringBoot

- clique o ícone de play no canto superior direito do Visual Studio Code.

Criando a tela para inserir chaves

- Abra o projeto front-end
- Entre na pasta pages
- Crie um arquivo InserirPage.svelte
- Copie o seguinte código:

```
<script>
    let chave = { nome: "", situacao: "disponivel", status: true };

    //Lista para mostrar as chaves
    let Listachaves = [];

    async function carregarChaves() {
        try {
            const response = await fetch(
                "http://localhost:8081/chaves/situacao/disponivel"
            ); // Adicione "http://" ao URL
            if (response.ok) {
                const chaves = await response.json();
                Listachaves = chaves;
                console.log(chaves); // Mude para "chaves" em vez de
"Listachaves"
            } else {
                console.error(
                    "Erro ao carregar as chaves:",
                    response.statusText
                );
            }
        } catch (error) {
            console.error("Erro ao carregar as chaves:", error);
        }
    }

    carregarChaves();

</script>
<main>
    <h2>Lista de Chaves Disponíveis:</h2>
    <ul>
        {#each Listachaves as chave}
            {#if chave.situacao == "disponivel"}
                <li>🗝{chave.nome} - Situação: {chave.situacao} ✔</li>
            {:else}
                <li>🗝{chave.nome} - Situação: {chave.situacao} ⛔</li>
            {/if}
        {/each}
    </ul>
</main>
<style>
```

```
    main {
        text-align: center;
        justify-content: center;
        display: block;
        background-color: rgb(34, 40, 49);
        color: rgb(238, 238, 238);
        overflow: auto;
    }

    h2 {
        font-size: 2.5em;
        font-family: "Courier New", Courier, monospace;
        color: rgb(214, 90, 49);
        font-weight: bolder;
    }

    button {
        background-color: rgb(214, 90, 49);
        color: rgb(238, 238, 238);
        padding: 1vh 2vh;
        border-radius: 5%;
        cursor: pointer;
        margin: 1vh;
        transition: background-color 0.3s;
        font-size: 1.5em;
    }

    button:hover {
        background-color: rgb(57, 62, 70);
    }

    label {
        margin-bottom: 5vh;
        font-family: Arial, Helvetica, sans-serif;
        font-weight: bold;
        font-size: 1.3em;
    }

    input {
        padding: 1vh;
        margin: 3vh;
        border: 5px solid rgb(93, 104, 122);
        border-radius: 4px;
    }
    li {
        list-style: none;
        margin: 2vh;
        font-size: 2em;
    }
</style>
```

iniciando o projeto Svelte

- Abra o terminal vá até a pasta frontend utilizando o cd e digite:
  npm run dev