Documentação do Sistema de Gestão de Horários e Alocações de Salas

Descrição Geral

Este sistema foi desenvolvido para gerenciar horários e alocações de salas em uma instituição educacional. Ele permite o carregamento de dados de salas e horários a partir de arquivos CSV, a visualização de disponibilidade de salas, a sugestão e substituição de alocações, e a criação de mapas de calor que mostram a ocupação de salas ao longo do tempo.

**Estrutura do Código-Salas.js**

Variáveis Principais

let table;: Referência para a tabela de dados.

let roomData = [];: Dados das salas.

let scheduleData = [];: Dados dos horários.

Eventos de Inicialização

O evento DOMContentLoaded garante que o código será executado após o carregamento completo do DOM.

Configuração dos manipuladores de eventos para carregamento de dados, verificação de horários pré-selecionados, e carregamento de sugestões.

Carregamento de Dados

loadRoomData(): Carrega dados das salas a partir de um arquivo CSV.

loadScheduleData(): Carrega dados de horários a partir de um arquivo CSV.

checkPreselectedSchedule(): Verifica se há dados de horários pré-selecionados no localStorage.

loadSuggestions(): Carrega sugestões de alocações a partir de um arquivo CSV.

Manipulação de Arquivos CSV

handleFileSelect2(event): Manipula a seleção de arquivo e processa o arquivo CSV.

parseFile2(file): Lê e processa o arquivo CSV carregado.

parseCSV2(csvData): Processa dados CSV em um formato utilizável.

Disponibilidade de Salas

checkAvailability(): Verifica a disponibilidade das salas com base na data e hora fornecidas, filtrando os horários relevantes.

updateTableWithAvailability(relevantSchedules): Atualiza a tabela com salas disponíveis.

Criação e Atualização da Tabela

createTable(headers, data): Cria uma tabela utilizando a biblioteca Tabulator, configurando colunas dinamicamente baseadas nos cabeçalhos do CSV.

Modais para Substituição e Adição de Aula

openModal(): Abre o modal para substituição de aulas.

closeModal(): Fecha o modal para substituição de aulas.

confirmSubstitution(rowData): Preenche os campos do modal com os dados da linha selecionada e abre o modal.

openAddClassModal(): Abre o modal para adição de nova aula.

closeAddClassModal(): Fecha o modal para adição de nova aula.

Submissão de Substituição e Nova Aula

submitSubstitution(event): Submete os dados do formulário de substituição e atualiza o CSV.

submitNewClass(event): Submete os dados do formulário de adição de nova aula e atualiza o CSV.

suggestAllocation(form): Sugere alocação de sala com base nos dados fornecidos e salva a sugestão no CSV.

saveDataToServer(csvContent): Envia os dados CSV para o servidor.

Funções Auxiliares

convertTimeToMinutes(timeString): Converte hora no formato HH:MM para minutos.

formatTime(time): Garante que a hora seja formatada corretamente.

formatDate(dateStr): Formata a data como DD/MM/AAAA.

calculateWeekOfYear(dateStr): Calcula a semana do ano para uma data.

calculateSemesterWeek(dateStr, semesterStartStr): Calcula a semana do semestre para uma data.

fetchHighestId(): Busca o maior ID atual no arquivo CSV.

addToScheduleCSV(newRow): Adiciona uma nova linha ao arquivo CSV de horários.

Criação de Heatmaps

criarHeatmapMatrizDias(): Cria um heatmap semanal mostrando a ocupação de salas.

criarHeatmapMatrizMensal(mes): Cria um heatmap mensal mostrando a ocupação de salas.

getColorForSalasOcupadas(salasOcupadas): Define a cor do quadrado no heatmap com base no número de salas ocupadas.

toggleHeatmap(): Alterna a visualização do heatmap semanal.

toggleHeatmapSelector(): Alterna a visualização do seletor de mês para o heatmap mensal.

Gerenciamento de Sugestões de Alocação

initializeSuggestionsTable(headers, data): Inicializa a tabela de sugestões de alocações.

substituir(): Realiza a substituição de uma alocação com base em uma sugestão.

DeleteAlocacaoByID(id): Remove uma alocação pelo ID.

fetchMaxSubID(): Busca o maior SubID no arquivo de sugestões de alocações.

Estrutura HTML e Modais

Criação de modais para substituição e adição de aula utilizando innerHTML e manipuladores de eventos para abrir e fechar os modais.

Criação de elementos de interface como tabelas, botões e formulários para interação com o usuário.

**Estrutura do Código- script.js**

Variáveis Principais

let table;: Referência para a tabela de dados.

Eventos de Inicialização

O evento DOMContentLoaded garante que o código será executado após o carregamento completo do DOM.

Configuração dos manipuladores de eventos para carregamento de dados, seleção de arquivos CSV e salvamento de arquivos no servidor.

Carregamento de Dados

fetchScheduleFile(): Carrega o arquivo CSV de horários do servidor.

parseFile(file): Lê o arquivo CSV e carrega seus dados na tabela.

parseCSV(csvData): Processa dados CSV em um formato utilizável.

Manipulação de Arquivos CSV

handleFileSelect(event): Manipula a seleção de arquivos CSV, dispara a leitura e processamento do arquivo.

parseFile(file): Lê o arquivo CSV e carrega seus dados na tabela.

parseCSV(csvData): Processa dados CSV em um formato utilizável.

Funções de Cálculo

calculateWeekOfYear(dateStr): Calcula a semana do ano com base na data fornecida.

calculateSemesterWeek(dateStr, semesterStartStr): Calcula a semana do semestre com base na data fornecida.

populateAdditionalColumns(data): Preenche colunas adicionais ('Semana do Ano', 'Semana do 1º Semestre', 'Semana do 2º Semestre') no conjunto de dados.

Criação e Atualização da Tabela

createTable(headers, data): Cria uma tabela utilizando a biblioteca Tabulator, configurando colunas dinamicamente baseadas nos cabeçalhos do CSV.

createColumnControls(headers): Cria controles de coluna para manipular a visibilidade das colunas.

toggleColumn(column): Alterna a visibilidade de uma coluna.

Funções de Filtro e Pesquisa

applyOrFilter(): Aplica um filtro "OU" para buscar dados na tabela com base no valor inserido.

orFilterFunction(data, filterParams): Define a função de filtro para a pesquisa "OU".

Funções de Exportação e Salvamento

uploadFile(): Exporta dados visíveis do Tabulator para CSV e envia para o servidor.

exportTabulatorToCSV(): Exporta os dados da tabela para o formato CSV.

saveFileToServer(): Salva o arquivo CSV no servidor.

fetchScheduleFile(): Carrega o arquivo CSV de horários do servidor.

Funções de Substituição e Remoção de Aulas

markSubstitution(data): Armazena as informações da aula que precisa de substituição e redireciona para a página de salas.

markSubstitutionByID(): Marca a substituição de uma aula pelo ID inserido e redireciona para a página de substituição.

DeleteAulaByID(): Apaga a aula pelo ID inserido.

saveUpdatedCSV(updatedCSV): Salva o arquivo CSV atualizado no servidor.

Estrutura HTML e Modais

Criação de modais para substituição e adição de aula utilizando innerHTML e manipuladores de eventos para abrir e fechar os modais.

Criação de elementos de interface como tabelas, botões e formulários para interação com o usuário.

Conclusão

Este sistema integra várias funcionalidades para facilitar a gestão de horários e alocações de salas, utilizando arquivos CSV para carregar e salvar dados, e bibliotecas como Tabulator para visualização e manipulação dos dados. A estrutura modular do código permite fácil manutenção e expansão do sistema conforme necessário.

Documentação do Sistema Backend para Gerenciamento de Horários e Alocações de Salas

Descrição Geral

Este sistema backend foi desenvolvido para gerenciar a carga, download e armazenamento de arquivos CSV contendo informações sobre horários e alocações de salas em uma instituição educacional. Ele inclui endpoints para carregar, baixar e armazenar arquivos, assim como para servir páginas HTML para visualização e interação com os dados.

Estrutura do Código- WebControler

Dependências

org.springframework.http.HttpStatus

org.springframework.http.ResponseEntity

org.springframework.stereotype.Controller

org.springframework.web.bind.annotation.GetMapping

org.springframework.web.bind.annotation.PostMapping

org.springframework.web.bind.annotation.RequestBody

org.springframework.web.bind.annotation.RequestParam

org.springframework.web.multipart.MultipartFile

org.apache.commons.csv.CSVParser

org.apache.commons.csv.CSVFormat

org.apache.commons.csv.CSVRecord

org.springframework.core.io.Resource

org.springframework.core.io.UrlResource

org.springframework.core.io.InputStreamResource

org.springframework.http.HttpHeaders

org.springframework.http.MediaType

java.io.IOException

java.nio.file.Files

java.nio.file.Path

java.nio.file.Paths

java.nio.charset.StandardCharsets

java.nio.file.StandardCopyOption

java.nio.file.StandardOpenOption

java.util.List

java.net.MalformedURLException

org.slf4j.Logger

org.slf4j.LoggerFactory

Controlador Web

WebController: Controlador principal que lida com as requisições HTTP para carregar, baixar e armazenar arquivos CSV, e para servir páginas HTML.

Mapeamento de Páginas HTML

@GetMapping("/"): Mapeia a página inicial (hub.html).

@GetMapping("/horarios"): Mapeia a página de horários (horarios.html).

@GetMapping("/salas"): Mapeia a página de salas (salas.html).

Endpoints para Upload de Arquivos CSV

@PostMapping("/upload-horarios"): Endpoint para upload de arquivos CSV de horários.

@PostMapping("/upload-salas"): Endpoint para upload de arquivos CSV de salas.

Endpoints para Download de Arquivos CSV

@GetMapping("/HorarioDeExemploAtualizado.csv"): Endpoint para download do arquivo CSV de horários.

@GetMapping("/caracterizacaodasSalas.csv"): Endpoint para download do arquivo CSV de caracterização das salas.

@GetMapping("/HorarioParaTestesV2.csv"): Endpoint para download do arquivo CSV de horários de teste (versão 2).

@GetMapping("/HorarioParaTestes.csv"): Endpoint para download do arquivo CSV de horários de teste (versão 1).

@GetMapping("/sugestaoAlocacoes.csv"): Endpoint para download do arquivo CSV de sugestões de alocações.

Endpoint para Salvamento de Dados CSV

@PostMapping("/save-csv"): Endpoint para salvar dados CSV recebidos no corpo da requisição.

Estrutura de Arquivos CSV

caracterizacaodasSalas.csv: Contém a caracterização das salas.

HorarioDeExemploAtualizado.csv: Contém os horários atualizados.

sugestaoAlocacoes.csv: Contém as sugestões de alocações.

Padrões de Resposta

ResponseEntity: Utilizado para criar respostas HTTP com corpo, cabeçalhos e status personalizados.

Resource: Utilizado para representar arquivos que podem ser baixados.

Tratamento de Erros

Uso de logs (Logger) para registrar erros ao acessar ou criar arquivos.

Verificação de existência e legibilidade dos arquivos antes de servir os mesmos.

Respostas HTTP apropriadas (e.g., HttpStatus.BAD\_REQUEST, HttpStatus.INTERNAL\_SERVER\_ERROR) em caso de erros.

Conclusão

Este sistema backend integra várias funcionalidades para gerenciar arquivos CSV que contêm informações críticas sobre horários e alocações de salas, suportando operações de upload e download de arquivos e servindo páginas HTML para interação com os dados. A estrutura modular do código permite fácil manutenção e expansão do sistema conforme necessário.