FileUpload Angular

Document.component.html

|  |
| --- |
| <br><br><br><br> |
|  | <div class="card bg-light text-dark text-center container"> |
|  | <div class="card-body"> |
|  | <h1><i class="fas fa-file-upload"></i> Upload Documents</h1> |
|  | </div> |
|  | </div> |
|  | <div class="container"> |
|  | <div class="row "> |
|  | <div class="col-md-4 py-5 bg-primary text-white text-center"> |
|  | <div class="card-body"> |
|  | <img src="http://www.ansonika.com/mavia/img/registration\_bg.svg" style="width:30%"> |
|  | <h2 class="py-3">Upload Documents</h2> |
|  | <p>Supported Documents Type Can be (pdf,jpg,jpeg,png) </p> |
|  | </div> |
|  | </div> |
|  | <div class="col-md-8 py-5 border"> |
|  | <h4 class="pb-4">Please Upload Necessary Documents</h4> |
|  | <form #DocumentForm="ngForm" (ngSubmit)="docc\_api(pancard.files,voterId.files,salarySlip.files,loa.files,noc.files,agreement.files)"> |
|  | <div class="row justify-content-center"> |
|  | <div class="form-group col-lg-2"> |
|  | <label for="PancardFile" class="form-label"><h5>PAN CARD</h5></label> |
|  | </div> |
|  | <div class="form-group col-lg-7"> |
|  | <input #pancard type="file" class="form-control" id="pancard" name="pancard" required> |
|  | </div> |
|  | </div> |
|  | <div class="row justify-content-center"> |
|  | <div class="form-group col-lg-2"> |
|  | <label for="VoterFile" class="form-label"><h5>VOTER-ID</h5></label> |
|  | </div> |
|  | <div class="form-group col-lg-7"> |
|  | <input #voterId type="file" class="form-control" id="voterId" name="voterId" required> |
|  | </div> |
|  | </div> |
|  | <div class="row justify-content-center"> |
|  | <div class="form-group col-lg-2"> |
|  | <label for="SalaryFile" class="form-label"><h5>SALARY SLIP</h5></label> |
|  | </div> |
|  | <div class="form-group col-lg-7"> |
|  | <input #salarySlip class="form-control" type="file" id="salarySlip" name="salarySlip" required> |
|  | </div> |
|  | </div> |
|  | <div class="row justify-content-center"> |
|  | <div class="form-group col-lg-2"> |
|  | <label for="LoaFile" class="form-label"><h5>LOA</h5></label> |
|  | </div> |
|  | <div class="form-group col-lg-7"> |
|  | <input #loa class="form-control" type="file" id="loa" name="loa" required> |
|  | </div> |
|  | </div> |
|  | <div class="row justify-content-center"> |
|  | <div class="form-group col-lg-2"> |
|  | <label for="NocFile" class="form-label"><h5>NOC</h5></label> |
|  | </div> |
|  | <div class="form-group col-lg-7"> |
|  | <input #noc class="form-control" type="file" id="noc" name="noc" required> |
|  | </div> |
|  | </div> |
|  | <div class="row justify-content-center"> |
|  | <div class="form-group col-lg-2"> |
|  | <label for="AgreementFile" class="form-label"><h5>AGREEMENT</h5></label> |
|  | </div> |
|  | <div class="form-group col-lg-7"> |
|  | <input #agreement class="form-control" type="file" id="agreement" name="agreement" required> |
|  | </div> |
|  | </div> |
|  | <br> |
|  | <div class="row justify-content-center"> |
|  | <div class="container text-center"> |
|  | <div class="row"> |
|  | <div class="col-md-6 py-3"> |
|  | <button type="submit" id="submit" name="submit" class="btn btn-lg btn-primary">Submit</button> |
|  | </div> |
|  | <div class="col-md-6 py-3"> |
|  | <form> |
|  | <button type="submit" id="cancel" name="cancel" class="btn btn-lg btn-danger" routerLink="/">Cancel</button> |
|  | </form> |
|  | </div> |
|  | <div class="alert alert-info mt-2" \*ngIf="msg">{{msg}}</div> |
|  | </div> |
|  | </div> |
|  | </div> |
|  | </form> |
|  | </div> |
|  | </div> |
|  | </div> |
|  |  |
|  | <hr> |
|  | <br><br><br><br> |
|  |  |
|  | <hr> |
|  |  |
|  | <!-- |
|  |  |
|  | <form #DocumentbyIDForm="ngForm" (ngSubmit)="get\_apibyID(DocumentbyIDForm.value)"> |
|  | <input type="text" id="dId" name="dId" [(ngModel)]="document.dId"> |
|  | <input type="submit"> |
|  | </form> |
|  | {{documents | json}} |
|  |  |
|  | <h1>Test Documents API</h1> |
|  | <div class="row"> |
|  | <div class="col-6 offset-3"> |
|  | <button type="button" class="btn btn-outline-primary btn-lg mt-5 ml-5" (click)="get\_api()">GET</button> |
|  | <button type="button" class="btn btn-outline-success btn-lg mt-5 ml-5" (click)="btn\_post()">POST</button> |
|  | <button type="button" class="btn btn-outline-dark btn-lg mt-5 ml-5" (click)="btn\_put()">PUT</button> |
|  | <button type="button" class="btn btn-outline-warning btn-lg mt-5 ml-5" (click)="btn\_delete()">DELETE</button> |
|  | <button type="button" class="btn btn-outline-danger btn-lg mt-5 ml-5" (click)="error\_api()">REGISTER</button> |
|  | </div> |
|  | </div> |
|  | <h1>Display Documents Tesst Data</h1> |
|  | <div class="row mt-5" \*ngIf="flag\_get"> |
|  | <div class="col-6 offset-3"> |
|  | <div class="card"> |
|  | <div class="card-body"> |
|  | <h4 class="card-title">Documents Data</h4> |
|  | <h6 class="card-subtitle mb-2 text-muted">Details of all Documents.</h6> |
|  | <hr> |
|  | <div class="card-text m-2 border border-warning p-2" \*ngFor="let item of documents" style="display: inline-block;"> |
|  | <h1>{{ item.cid }}</h1> |
|  | </div> |
|  | </div> |
|  | </div> |
|  | </div> |
|  | </div> |
|  |  |
|  | <h1>Test Documents API</h1> |
|  | --> |

**Document.component.ts**

|  |
| --- |
| import { Component, EventEmitter, OnInit, Output } from '@angular/core'; |
|  | import { DocumentsService } from '../documents.service'; |
|  | import { DOCUMENT } from '../Models/documents'; |
|  | import { error } from '@angular/compiler/src/util'; |
|  | import { HttpClient, HttpEventType, HttpHeaders } from '@angular/common/http'; |
|  | import { ActivatedRoute, Router } from '@angular/router'; |
|  | import { ApplicationService } from '../application.service'; |
|  | import { APPLICATION } from '../Models/application'; |
|  |  |
|  | @Component({ |
|  | selector: 'app-documents', |
|  | templateUrl: './documents.component.html', |
|  | styleUrls: ['./documents.component.css'] |
|  | }) |
|  | export class DocumentsComponent implements OnInit { |
|  |  |
|  | document:DOCUMENT={ |
|  | pancard:"", |
|  | voterId : "", |
|  | salarySlip : "", |
|  | loa : "", |
|  | noc : "", |
|  | agreement : "", |
|  | lId : 2, |
|  | prId : 2, |
|  | inId : 2, |
|  | cid : 1009, |
|  | }; |
|  |  |
|  | constructor(private http:HttpClient,private obja:ApplicationService,private obj:DocumentsService,private route: ActivatedRoute,private router:Router) { |
|  | //this.cuid = this.route.snapshot.params.id; |
|  | //console.log(this.cuid); |
|  | } |
|  |  |
|  | ngOnInit(): void { |
|  |  |
|  | this.document.cid=(this.route.snapshot.paramMap.get('cid')) as unknown as number; |
|  | this.document.inId=(this.route.snapshot.paramMap.get('inId')) as unknown as number; |
|  | this.document.prId=(this.route.snapshot.paramMap.get('prId')) as unknown as number; |
|  | this.document.lId=(this.route.snapshot.paramMap.get('lId')) as unknown as number; |
|  | console.log(this.document); |
|  | } |
|  |  |
|  | cuid:number=0; |
|  |  |
|  | documents:DOCUMENT[]=[]; |
|  |  |
|  | applications:APPLICATION[]=[]; |
|  |  |
|  | message:string=""; |
|  |  |
|  | //@Output() public onUploadFinished = new EventEmitter(); |
|  |  |
|  |  |
|  | id:number=9; //Using this Default for Primary key access |
|  |  |
|  | application:APPLICATION={ |
|  | loanStatus : "Sent for Verification", |
|  | lId : this.document.lId, |
|  | prId : this.document.prId, |
|  | inId : this.document.inId, |
|  | cid: this.document.cid, |
|  | aid : 2500, |
|  | } |
|  |  |
|  | //Other required variables. |
|  | msg:string=""; |
|  | u\_msg:string=""; |
|  | d\_msg:string=""; |
|  | e\_msg:string=""; |
|  |  |
|  | flag\_get:boolean=false; |
|  | flag\_post:boolean=false; |
|  | flag\_put:boolean=false; |
|  | flag\_delete:boolean=false; |
|  | flag\_register:boolean=false; |
|  |  |
|  | update\_id:number=0; |
|  | delete\_id:number=0; |
|  |  |
|  |  |
|  | docc\_api = (data1:any,data2:any,data3:any,data4:any,data5:any,data6:any) => |
|  | { |
|  | console.log() |
|  | let temp:any; |
|  | let f1:File[]=data1; |
|  | let f2:File[]=data2; |
|  | let f3:File[]=data3; |
|  | let f4:File[]=data4; |
|  | let f5:File[]=data5; |
|  | let f6:File[]=data6; |
|  |  |
|  | let fname1:any; |
|  | let fname2:any; |
|  | let fname3:any; |
|  | let fname4:any; |
|  | let fname5:any; |
|  | let fname6:any; |
|  |  |
|  | const formData1 = new FormData(); |
|  | const formData2 = new FormData(); |
|  | const formData3 = new FormData(); |
|  | const formData4 = new FormData(); |
|  | const formData5 = new FormData(); |
|  | const formData6 = new FormData(); |
|  |  |
|  | fname1=Array.from(f1).map((file, index) => { |
|  | fname1=this.document.lId+file.name; |
|  | formData1.append('file'+index, file,fname1); |
|  | return fname1; |
|  | }); |
|  |  |
|  | this.http.post('https://localhost:44302/api/Documents/PostDoc',formData1).subscribe(event => { |
|  | }); |
|  |  |
|  | fname2=Array.from(f2).map((file, index) => { |
|  | fname2=this.document.lId+file.name; |
|  | formData2.append('file'+index, file,fname2); |
|  | return fname2; |
|  | }); |
|  |  |
|  | this.http.post('https://localhost:44302/api/Documents/PostDoc',formData2).subscribe(event => { |
|  | }); |
|  |  |
|  | fname3=Array.from(f3).map((file, index) => { |
|  | fname3=this.document.lId+file.name; |
|  | formData3.append('file'+index, file,fname3); |
|  | return fname3; |
|  | }); |
|  |  |
|  | this.http.post('https://localhost:44302/api/Documents/PostDoc',formData3).subscribe(event => { |
|  | }); |
|  |  |
|  | fname4=Array.from(f4).map((file, index) => { |
|  | fname4=this.document.lId+file.name; |
|  | formData4.append('file'+index, file,fname4); |
|  | return fname4; |
|  | }); |
|  |  |
|  | this.http.post('https://localhost:44302/api/Documents/PostDoc',formData4).subscribe(event => { |
|  | }); |
|  |  |
|  | fname5=Array.from(f5).map((file, index) => { |
|  | fname5=this.document.lId+file.name; |
|  | formData5.append('file'+index, file,fname5); |
|  | return fname5; |
|  | }); |
|  |  |
|  | this.http.post('https://localhost:44302/api/Documents/PostDoc',formData5).subscribe(event => { |
|  | }); |
|  |  |
|  | fname6=Array.from(f6).map((file, index) => { |
|  | fname6=this.document.lId+file.name; |
|  | formData6.append('file'+index, file,fname6); |
|  | return fname6; |
|  | }); |
|  |  |
|  | this.http.post('https://localhost:44302/api/Documents/PostDoc',formData6).subscribe(event => { |
|  | }); |
|  |  |
|  |  |
|  | let str="https://localhost:44302/Resources/UDocuments"; |
|  | this.document.pancard="https://localhost:44302/Resources/UDocuments/"+fname1; |
|  | this.document.voterId="https://localhost:44302/Resources/UDocuments/"+fname2; |
|  | this.document.salarySlip="https://localhost:44302/Resources/UDocuments/"+fname3; |
|  | this.document.loa="https://localhost:44302/Resources/UDocuments/"+fname4; |
|  | this.document.noc="https://localhost:44302/Resources/UDocuments/"+fname5; |
|  | this.document.agreement="https://localhost:44302/Resources/UDocuments/"+fname6; |
|  |  |
|  | console.log(fname1,fname2,fname3,fname4,fname5,fname6); |
|  | this.obj.UploadDocuments(this.document).subscribe(data=>{ |
|  | //Logging the response received from web api. |
|  | console.log(data); |
|  | console.log(this.document); |
|  |  |
|  | this.application.loanStatus="Sent for Verification", |
|  | this.application.lId = this.document.lId, |
|  | this.application.prId = this.document.prId, |
|  | this.application.inId = this.document.inId, |
|  | this.application.cid = this.document.cid, |
|  | this.application.aid = 2500, |
|  |  |
|  |  |
|  | this.obja.CreateApplication(this.application).subscribe(adata=>{ |
|  |  |
|  | this.router.navigate(['/TrackLoan'+'/'+adata.cid+'/'+adata.inId+'/'+adata.prId+'/'+adata.lId+'/'+data.dId+'/'+adata.applicationNumber]); |
|  |  |
|  | }); |
|  |  |
|  | }); |
|  | } |
|  |  |
|  |  |
|  |  |
|  | doc\_api = (data:any) => |
|  | { |
|  | //console.log(data); |
|  | let filesToUpload : File[] = data; |
|  | const formData = new FormData(); |
|  |  |
|  | Array.from(filesToUpload).map((file, index) => { |
|  | return formData.append('file'+index, file, file.name); |
|  | }); |
|  |  |
|  | this.http.post('https://localhost:44302/api/Documents', formData).subscribe(event => { |
|  | console.log(event); |
|  | }); |
|  |  |
|  | } |
|  |  |
|  |  |
|  |  |
|  | //Upload Files Dynamic |
|  | public uploadFile = (files:any) => { |
|  | if (files.length === 0) { |
|  | return; |
|  | } |
|  | let fileToUpload = <File>files[0]; |
|  | console.log(fileToUpload); |
|  | const formData = new FormData(); |
|  | formData.append('pancard', fileToUpload, fileToUpload.name); |
|  | this.http.post('https://localhost:44302/api/Documents', formData, {reportProgress: true, observe: 'events'}) |
|  | .subscribe(event => { |
|  | console.log(event); |
|  | }); |
|  |  |
|  | } |
|  |  |
|  |  |
|  |  |
|  |  |
|  | postt\_api(data:any):void //Create New Use POST |
|  | { |
|  | console.log(data); |
|  | data.lId = this.document.lId; |
|  | data.prId = this.document.prId; |
|  | data.inId = this.document.inId; |
|  | data.cid = this.document.cid; |
|  | console.log(data); |
|  | console.log(this.document); |
|  | } |
|  |  |
|  | get\_api():void //Get All Documents Details |
|  | { |
|  | this.obj.getDocuments().subscribe(data=>{ |
|  | this.documents=data; |
|  | this.flag\_get=true;this.flag\_post=false;this.flag\_put=false;this.flag\_delete=false;this.flag\_register=false; |
|  | //Logging the response recieved from web api. |
|  | console.log(this.documents); |
|  | }); |
|  | } |
|  |  |
|  |  |
|  | get\_apibyID(data:any):void //Get Document Details by ID |
|  | { |
|  | console.log(data); |
|  | console.log(typeof(data.dId)); |
|  | this.id = data.dId; |
|  | this.obj.getDocumentsbyID(this.id).subscribe(data=>{ |
|  | this.documents=data; |
|  | //Logging the response recieved from web api. |
|  | console.log(this.documents); |
|  | }); |
|  | } |
|  |  |
|  | post\_api(data:any):void //Create New Use POST |
|  | { |
|  | console.log(data); |
|  | console.log(this.document); |
|  | data.lId = this.document.lId; |
|  | data.prId = this.document.prId; |
|  | data.inId = this.document.inId; |
|  | data.cid = this.document.cid; |
|  | this.obj.UploadDocuments(data).subscribe(data=>{ |
|  | this.msg="Successfully created "+data.dId; |
|  | //Logging the response received from web api. |
|  | console.log(data); |
|  | console.log(this.document); |
|  |  |
|  | }); |
|  | } |
|  |  |
|  |  |
|  | put\_api(id:number,data:any):void //UPDATE Existing Use PUT |
|  | { |
|  | this.obj.UpdateDocuments(id,data).subscribe(data=>{ |
|  | this.u\_msg="Successfully Updated Documents"+id; |
|  | console.log(data); |
|  | }) |
|  |  |
|  | } |
|  |  |
|  | delete\_api(id:number):void //DELETE Existing Use Delete |
|  | { |
|  | this.obj.DeleteDocument(id).subscribe(data=>{ |
|  | this.d\_msg="Successfully Deleted Document Record"+id; |
|  | console.log(data); |
|  | }) |
|  |  |
|  | } |
|  |  |
|  | error\_api():void |
|  | { |
|  | this.flag\_get=false; |
|  | this.flag\_post=false; |
|  | this.flag\_put=false; |
|  | this.flag\_delete=false; |
|  | this.flag\_register=true; |
|  |  |
|  | this.obj.register().subscribe((data)=>{ |
|  | console.log(data); |
|  | }, |
|  | (error)=>{ |
|  | this.e\_msg=error; |
|  | } |
|  | ); |
|  | } |
|  |  |
|  | btn\_post():void |
|  | { |
|  | this.flag\_get=false; |
|  | this.flag\_post=true; |
|  | this.flag\_put=false; |
|  | this.flag\_delete=false; |
|  | this.flag\_register=false; |
|  | } |
|  |  |
|  | btn\_put():void |
|  | { |
|  | this.flag\_get=false; |
|  | this.flag\_post=false; |
|  | this.flag\_put=true; |
|  | this.flag\_delete=false; |
|  | this.flag\_register=false; |
|  | } |
|  |  |
|  | btn\_delete():void |
|  | { |
|  | this.flag\_get=false; |
|  | this.flag\_post=false; |
|  | this.flag\_put=false; |
|  | this.flag\_delete=true; |
|  | this.flag\_register=false; |
|  | } |
|  |  |
|  |  |
|  | } |

**DocumentService.ts**

|  |
| --- |
| import { Injectable } from '@angular/core'; |
|  | import { HttpClient, HttpErrorResponse, HttpHeaders } from '@angular/common/http'; |
|  | import { DOCUMENT } from './Models/documents'; |
|  | //import { Observable } from 'rxjs/internal/Observable'; |
|  | import { Observable, throwError } from 'rxjs'; |
|  | import {catchError} from 'rxjs/operators'; |
|  |  |
|  |  |
|  | @Injectable({ |
|  | providedIn: 'root' |
|  | }) |
|  |  |
|  | export class DocumentsService { |
|  |  |
|  | //Creating an instance of HttpClient inside the constructor. |
|  | constructor(private http:HttpClient) { } |
|  |  |
|  | //Variable to store the request URL for accessing API. |
|  | req:string="https://localhost:44302/api/Documents"; |
|  |  |
|  |  |
|  | //Method to get the list of all Dcouments from the API. |
|  | getDocuments():Observable<DOCUMENT[]> |
|  | { |
|  | return this.http.get<DOCUMENT[]>(this.req); |
|  | } |
|  |  |
|  | getDocumentsbyID(id:number):Observable<DOCUMENT[]> |
|  | { |
|  | return this.http.get<DOCUMENT[]>(this.req+"/"+id,{ |
|  | headers:new HttpHeaders({ |
|  | 'Content-Type':'application/json;charset=UTF-8', |
|  | 'Access-Control-Allow-Origin':'\*', |
|  | 'Access-Control-Allow-Method':'\*' |
|  | }) |
|  | }); |
|  | } |
|  |  |
|  | //Method to Create and Upload a new Document Dynamic. |
|  |  |
|  |  |
|  |  |
|  |  |
|  | //Method to Create and Upload a new Document. |
|  | UploadDocuments(document:DOCUMENT):Observable<DOCUMENT> |
|  | { |
|  | return this.http.post<DOCUMENT>(this.req+"/Upload",document,{ |
|  | headers:new HttpHeaders({ |
|  | 'Content-Type':'application/json;charset=UTF-8', |
|  | 'Access-Control-Allow-Origin':'\*', |
|  | 'Access-Control-Allow-Method':'\*' |
|  | }) |
|  | }); |
|  | } |
|  |  |
|  | //Method to Update an Existing Document. |
|  | UpdateDocuments(id:number,document:DOCUMENT):Observable<any> |
|  | { |
|  | return this.http.put<any>(this.req+"/"+id,document,{ |
|  | headers:new HttpHeaders({ |
|  | 'Content-Type':'application/json;charset=UTF-8', |
|  | 'Access-Control-Allow-Origin':'\*', |
|  | 'Access-Control-Allow-Method':'\*' |
|  | }) |
|  | }); |
|  | } |
|  |  |
|  | //Method to Delete an Existing Dcoument. |
|  | DeleteDocument(id:number):Observable<any> |
|  | { |
|  | return this.http.delete<any>(this.req+"/"+id,{ |
|  | headers:new HttpHeaders({ |
|  | 'Content-Type':'application/json;charset=UTF-8', |
|  | 'Access-Control-Allow-Origin':'\*', |
|  | 'Access-Control-Allow-Method':'\*' |
|  | }) |
|  | }); |
|  | } |
|  |  |
|  | //Method to test Error Handling. |
|  | register():Observable<any> |
|  | { |
|  | //Giving incorrect URL. |
|  | return this.http.get<any>('https://localhost:44302/api/Documents') |
|  | .pipe(catchError(this.manageError)); |
|  | } |
|  |  |
|  | //Method to handle errors. |
|  | private manageError(err\_response:HttpErrorResponse) |
|  | { |
|  | if(err\_response.error instanceof ErrorEvent) |
|  | console.error('Client Side Error:',err\_response.error.message); |
|  | else |
|  | console.error('Server Side Error:',err\_response); |
|  |  |
|  | return throwError('There is a little problem while processing your request.Sorry for the inconvenience'); |
|  |  |
|  | } |
|  |  |
|  | } |

**API**

**Model**

|  |
| --- |
| public partial class Document |
|  | { |
|  | public int DocumentId { get; set; } |
|  | public string Username { get; set; } |
|  | public byte[] DocImage { get; set; } |
|  | } |

Document.cs

**DocumentController**

|  |
| --- |
| using Microsoft.AspNetCore.Http; |
|  | using Microsoft.AspNetCore.Mvc; |
|  | using Microsoft.EntityFrameworkCore; |
|  | using ngToASP.FinanceModel; |
|  | using System; |
|  | using System.Web; |
|  | using System.Collections.Generic; |
|  | using System.IO; |
|  | using System.Linq; |
|  | using System.Threading.Tasks; |
|  |  |
|  | namespace ngToASP.Controllers |
|  | { |
|  | [Route("api/[controller]")] |
|  | [ApiController] |
|  |  |
|  | public class DocumentsController : ControllerBase |
|  | { |
|  |  |
|  |  |
|  | private readonly ProjectGladiatorContext \_context; |
|  |  |
|  | public DocumentsController(ProjectGladiatorContext context) |
|  | { |
|  | \_context = context; |
|  | } |
|  |  |
|  | // GET: api/Documents |
|  | [HttpGet] |
|  | public async Task<ActionResult<IEnumerable<Document>>> GetDocuments() |
|  | { |
|  | return await \_context.Documents.ToListAsync(); |
|  | } |
|  |  |
|  | // GET: api/Documents/5 |
|  | [HttpGet("{id}")] |
|  | public async Task<ActionResult<Document>> GetDocument(int id) |
|  | { |
|  | var document = await \_context.Documents.FindAsync(id); |
|  |  |
|  | if (document == null) |
|  | { |
|  | return NotFound(); |
|  | } |
|  |  |
|  | return document; |
|  | } |
|  |  |
|  | // PUT: api/Documents/5 |
|  | // To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754 |
|  | [HttpPut("{id}")] |
|  | public async Task<IActionResult> PutDocument(int id, Document document) |
|  | { |
|  | if (id != document.DocumentId) |
|  | { |
|  | return BadRequest(); |
|  | } |
|  |  |
|  | \_context.Entry(document).State = EntityState.Modified; |
|  |  |
|  | try |
|  | { |
|  | await \_context.SaveChangesAsync(); |
|  | } |
|  | catch (DbUpdateConcurrencyException) |
|  | { |
|  | if (!DocumentExists(id)) |
|  | { |
|  | return NotFound(); |
|  | } |
|  | else |
|  | { |
|  | throw; |
|  | } |
|  | } |
|  |  |
|  | return NoContent(); |
|  | } |
|  |  |
|  | // POST: api/Documents |
|  | // To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754 |
|  | /\* [HttpPost] |
|  | public async Task<ActionResult<Document>> PostDocument(Document document) |
|  | { |
|  |  |
|  |  |
|  | \_context.Documents.Add(document); |
|  | await \_context.SaveChangesAsync(); |
|  |  |
|  | return CreatedAtAction("GetDocument", new { id = document.DocumentId }, document); |
|  | }\*/ |
|  |  |
|  | // DELETE: api/Documents/5 |
|  | [HttpDelete("{id}")] |
|  | public async Task<IActionResult> DeleteDocument(int id) |
|  | { |
|  | var document = await \_context.Documents.FindAsync(id); |
|  | if (document == null) |
|  | { |
|  | return NotFound(); |
|  | } |
|  |  |
|  | \_context.Documents.Remove(document); |
|  | await \_context.SaveChangesAsync(); |
|  |  |
|  | return NoContent(); |
|  | } |
|  |  |
|  | private bool DocumentExists(int id) |
|  | { |
|  | return \_context.Documents.Any(e => e.DocumentId == id); |
|  | } |
|  | [HttpPost] |
|  | public IActionResult UploadImage(IFormFile file) |
|  | { |
|  |  |
|  | if (file.Length > 0) |
|  | { |
|  | var fileName = Path.GetFileName(file.FileName); |
|  | using (FileStream stream = new FileStream(Path.Combine("Image", fileName), FileMode.Create)) |
|  |  |
|  | file.CopyTo(stream); |
|  | Document doc = new Document(); |
|  | doc.Username = fileName; |
|  |  |
|  | \_context.Documents.Add(doc); |
|  | \_context.SaveChanges(); |
|  |  |
|  | } |
|  | return Ok(); |
|  | } |
|  | } |
|  | } |

**Model**

Dbimg.cs

|  |
| --- |
| public partial class Dbimg |
|  | { |
|  | public int Iid { get; set; } |
|  | public string ImageName { get; set; } |
|  | public string ImageFile { get; set; } |
|  | } |

Dbimage.controller

|  |
| --- |
| using System; |
|  | using System.Collections.Generic; |
|  | using System.Linq; |
|  | using System.Threading.Tasks; |
|  | using Microsoft.AspNetCore.Http; |
|  | using Microsoft.AspNetCore.Mvc; |
|  | using Microsoft.EntityFrameworkCore; |
|  | using ngToASP.FinanceModel; |
|  |  |
|  | namespace ngToASP.Controllers |
|  | { |
|  | [Route("api/[controller]")] |
|  | [ApiController] |
|  | public class DbimgsController : ControllerBase |
|  | { |
|  | private readonly ProjectGladiatorContext \_context; |
|  |  |
|  | public DbimgsController(ProjectGladiatorContext context) |
|  | { |
|  | \_context = context; |
|  | } |
|  |  |
|  | // GET: api/Dbimgs |
|  | [HttpGet] |
|  | public async Task<ActionResult<IEnumerable<Dbimg>>> GetDbimgs() |
|  | { |
|  | return await \_context.Dbimgs.ToListAsync(); |
|  | } |
|  |  |
|  | // GET: api/Dbimgs/5 |
|  | [HttpGet("{id}")] |
|  | public async Task<ActionResult<Dbimg>> GetDbimg(int id) |
|  | { |
|  | var dbimg = await \_context.Dbimgs.FindAsync(id); |
|  |  |
|  | if (dbimg == null) |
|  | { |
|  | return NotFound(); |
|  | } |
|  |  |
|  | return dbimg; |
|  | } |
|  |  |
|  | // PUT: api/Dbimgs/5 |
|  | // To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754 |
|  | [HttpPut("{id}")] |
|  | public async Task<IActionResult> PutDbimg(int id, Dbimg dbimg) |
|  | { |
|  | if (id != dbimg.Iid) |
|  | { |
|  | return BadRequest(); |
|  | } |
|  |  |
|  | \_context.Entry(dbimg).State = EntityState.Modified; |
|  |  |
|  | try |
|  | { |
|  | await \_context.SaveChangesAsync(); |
|  | } |
|  | catch (DbUpdateConcurrencyException) |
|  | { |
|  | if (!DbimgExists(id)) |
|  | { |
|  | return NotFound(); |
|  | } |
|  | else |
|  | { |
|  | throw; |
|  | } |
|  | } |
|  |  |
|  | return NoContent(); |
|  | } |
|  |  |
|  | // POST: api/Dbimgs |
|  | // To protect from overposting attacks, see https://go.microsoft.com/fwlink/?linkid=2123754 |
|  | [HttpPost] |
|  | public async Task<ActionResult<Dbimg>> PostDbimg(Dbimg dbimg) |
|  | { |
|  | \_context.Dbimgs.Add(dbimg); |
|  | await \_context.SaveChangesAsync(); |
|  |  |
|  | return CreatedAtAction("GetDbimg", new { id = dbimg.Iid }, dbimg); |
|  | } |
|  |  |
|  | // DELETE: api/Dbimgs/5 |
|  | [HttpDelete("{id}")] |
|  | public async Task<IActionResult> DeleteDbimg(int id) |
|  | { |
|  | var dbimg = await \_context.Dbimgs.FindAsync(id); |
|  | if (dbimg == null) |
|  | { |
|  | return NotFound(); |
|  | } |
|  |  |
|  | \_context.Dbimgs.Remove(dbimg); |
|  | await \_context.SaveChangesAsync(); |
|  |  |
|  | return NoContent(); |
|  | } |
|  |  |
|  | private bool DbimgExists(int id) |
|  | { |
|  | return \_context.Dbimgs.Any(e => e.Iid == id); |
|  | } |
|  | } |
|  | } |