# Project

* Site-10-Api-Ng-upload-photos-cloudinary
  + Copied from: Site-09-Api-Ng-templateform-candeactivate-spinner-intercepter-caching
* For the “MySocialConnect-SPA” issue npm install command to install all packages
  + ng serve : to run the spa

# New & Updates

|  |  |
| --- | --- |
| New | Updates |
| Ng2-file-upload | /site/members/member-edit |
|  | /core/modules/shared.module.ts |
|  | /core/services/member.service.ts |
|  | /site/nav/nav.component |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Clear Database

If you want to clear database then drop it and recreate it

dotnet ef database drop

dotnet ef database update

# ng2-file-upload

<https://valor-software.com/ng2-file-upload/>

npm install ng2-file-upload

or

npm i ng2-file-upload

## /core/modules/shared.module.ts

Add FileUploadModule to imports and exports array

# /core/services

## member.service.ts

Add methods for setting main photo and deleting photo

  setMainPhoto(photoId: number){

    const url = this.helperService.replaceKeyValue(this.helperService.urlUserPhotoSetMain, this.helperService.keyPhotoId, photoId, "");

    this.helperService.logIfFrom(url, "memberService setMainPhoto");

    return this.httpClientService.put(url, {});

  }

  deletePhoto(photoId: number){

    const url = this.helperService.replaceKeyValue(this.helperService.urlUserPhotoDelete, this.helperService.keyPhotoId, photoId, "");

    this.helperService.logIfFrom(url, "memberService deletePhoto");

    return this.httpClientService.delete(url);

  }

# /site/nav

Add the user main photo to the nav

## nav.component.css

img {

    max-height: 50px;

    border: 2px solid white;

    display: inline;

}

## nav.component.html

        <!--logged in dropped down section-->

        <div class="dropdown" \*ngIf="isLoggedIn" dropdown>

          <!--add the main photo-->

          <img src="{{loggedInUser?.mainPhotoUrl || './assets/user.png'}}" alt="">

          <a class="dropdown-toggle text-light text-decoration-none ms-2" dropdownToggle>

# /site/members

## member-edit

### member-edit.html

Add the app-photo-editor to the edit photos tab

            <tab heading="Edit Photos">

                <app-photo-editor [member]="member"></app-photo-editor>

            </tab>

## photo-editor

### photo-editor.component.css

.nv-file-over{

    border: dotted 3px red;

}

input[type=file]{

    color: transparent;

}

### photo-editor.component.ts

import { Component, Input, OnDestroy, OnInit } from '@angular/core';

import { Subscription, take } from 'rxjs';

import { FileUploader } from 'ng2-file-upload';

import { ToastrService } from 'ngx-toastr';

import { HelperService } from '../../../core/services/helper.service';

import { AccountService } from '../../../core/services/account.service';

import { MemberService } from '../../../core/services/member.service';

import { LoggedInUserDto } from '../../../core/models-interfaces/logged-in-user-dto.model';

import { UserDto } from '../../../core/models-interfaces/user-dto.model';

import { PhotoDto } from '../../../core/models-interfaces/photo-dto.model';

import { AppConstants } from '../../../core/constants/app-constants';

@Component({

  selector: 'app-photo-editor',

  templateUrl: './photo-editor.component.html',

  styleUrls: ['./photo-editor.component.css']

})

export class PhotoEditorComponent implements OnInit, OnDestroy {

  @Input() member: UserDto | undefined;

  title = "PhotoEditor";

  user: LoggedInUserDto | undefined;

  /\* file uplaoder properties start \*/

  uploader: FileUploader | undefined;

  hasBaseDropZoneOver = false;

  /\* file uplaoder properties end \*/

  setPhotoMainSubscription!: Subscription;

  deletePhotoSubscriptoion!: Subscription;

  constructor(private helperService: HelperService, private toastr: ToastrService,

      private accountService: AccountService, private memberService: MemberService){

    //fill the logged in user, taking 1 so no need to unsubscribe

    this.accountService.currentLoggedInUser$.pipe(take(1)).subscribe({

      next: user => {

        if(user) this.user = user;

      }

    });

  }

  ngOnDestroy(): void {

    if(this.setPhotoMainSubscription) this.setPhotoMainSubscription.unsubscribe();

    if(this.deletePhotoSubscriptoion) this.deletePhotoSubscriptoion.unsubscribe();

  }

  ngOnInit(): void {

    this.printInfo();

    this.initilizeUploader();

  }

  //to set the drop zone, name must be exact

  fileOverBase(e: any){

    this.hasBaseDropZoneOver = e;

  }

  private printInfo(){

    this.helperService.logIf(`\*\*\*${this.title}\*\*\*`);

    this.helperService.logIf(`url: ${this.helperService.urlUserPhotoAdd}`);

    this.helperService.logIf(`user: ${this.user?.userName}`);

  }

  /\* initilizeUploader start \*/

  initilizeUploader(){

    const maxFileSize = 10 \* 1024 \* 1024; //10 mb image only

    //setup

    this.uploader = new FileUploader({

      url: this.helperService.urlUserPhotoAdd,

      authToken: `${AppConstants.Bearer}${this.user?.token}`,

      isHTML5: true,

      allowedFileType: ['image'], //jpeg, png, gif etc are allowed

      removeAfterUpload: true, //remove the file from drop zone

      autoUpload: false, //user will need to click the button

      maxFileSize: maxFileSize,

    });

    //we are using bearer token. Otherwise will need to adjust the cors configuration and allow credentials to go with file

    this.uploader.onAfterAddingFile = (file) => {

      file.withCredentials = false

    }

    //push the photo to the members.photos array after successful upload

    //it is making the users/add/photo call which is returning the added photo. Photo is in the response

    this.uploader.onSuccessItem = (item, response, status, headers) => {

      if(response){

        const photo = JSON.parse(response);

        this.member?.photos.push(photo);

        if(photo.isMain){

          if(this.user) this.user.mainPhotoUrl = photo.url;

          if(this.member) this.member.photoUrl = photo.url;

          this.accountService.setAndFireCurrentUser(this.user!);

        }

      }

    }

    //when the file is not added display error message

    this.uploader.onWhenAddingFileFailed = (item, filter) => {

      switch(filter.name){

        case 'fileSize': {

          const fileSizeError = `${item.name} is ${this.formatBytes(item.size)}. Max allowed is ${this.formatBytes(maxFileSize)}`;

          this.toastr.error(fileSizeError, `Error: ${filter.name}`);

          break;

        }

        default: {

          this.toastr.error(`Error while uploading file ${item.name}`, `Error: ${filter.name}`);

          break;

        }

      }

    }

  }

  /\* initilizeUploader end \*/

  private formatBytes(bytes:number, decimals?: number) {

    if (bytes == 0) return '0 Bytes';

    const k = 1024,

      dm = decimals || 2,

      sizes = ['Bytes', 'KB', 'MB', 'GB', 'TB', 'PB', 'EB', 'ZB', 'YB'],

      i = Math.floor(Math.log(bytes) / Math.log(k));

    return parseFloat((bytes / Math.pow(k, i)).toFixed(dm)) + ' ' + sizes[i];

  }

  onSetMainPhoto(photo: PhotoDto){

    this.setPhotoMainSubscription = this.memberService.setMainPhoto(photo.id).subscribe({

      next: () => {

        if(this.user && this.member) {

          this.user.mainPhotoUrl = photo.url;

          this.accountService.setAndFireCurrentUser(this.user);

          this.member.photoUrl = photo.url;

          this.member.photos.forEach(p => {

            if(p.isMain) p.isMain = false;

            if(p.id === photo.id) p.isMain = true;

          });

        }

      },

      error: e => {},

      complete: () => {}

    });

  }

  onDeletePhoto(photo: PhotoDto){

    this.deletePhotoSubscriptoion = this.memberService.deletePhoto(photo.id).subscribe({

      next: () => {

        //remove the photos from the member

        if(this.member && this.member.photos)

          this.member.photos = this.member.photos.filter(x => x.id !== photo.id);

      },

      error: e => {},

      complete: () => {}

    })

  }

}

### photo-editor.component.html

<div class="row mb-2" \*ngIf="member">

    <div class="col-2 mb-1" \*ngFor="let photo of member?.photos">

        <img src="{{photo.url}}" alt="Photo of user" class="img-thumbnail mb-1">

        <div class="text-center">

            <button class="btn btn-sm"

                [disabled]="photo.isMain"

                (click)="onSetMainPhoto(photo)"

                [ngClass]="photo.isMain ? 'btn-success active' : 'btn-outline-success'">

                Main</button>

            <button class="btn-sm btn-danger ms-2" \*ngIf="!photo.isMain"

            (click)="onDeletePhoto(photo)"><i class="fa fa-trash"></i></button>

        </div>

    </div>

</div>

<hr \*ngIf="member && member.photos">

<!--photo uploader start-->

<!--copied from https://valor-software.com/ng2-file-upload/ -->

<div class="row">

    <div class="col-3">

        <h3>Add photos</h3>

        <div ng2FileDrop

             [ngClass]="{'nv-file-over': hasBaseDropZoneOver}"

             (fileOver)="fileOverBase($event)"

             [uploader]="uploader"

             class="card bg-faded p-3 text-center mb-3 my-drop-zone" >

             <i class="fa fa-upload fa-3x"></i>

            Drop photos here

        </div>

        Multiple

        <input type="file" ng2FileSelect [uploader]="uploader" multiple  /><br/>

        Single

        <input type="file" ng2FileSelect [uploader]="uploader" />

    </div>

    <div class="col-9" style="margin-bottom: 40px" \*ngIf="uploader?.queue?.length">

        <h3>Upload queue</h3>

        <p>Queue length: {{ uploader?.queue?.length }}</p>

        <table class="table">

            <thead>

            <tr>

                <th width="50%">Name</th>

                <th>Size</th>

            </tr>

            </thead>

            <tbody>

            <tr \*ngFor="let item of uploader?.queue">

                <td><strong>{{ item?.file?.name }}</strong></td>

                <td \*ngIf="uploader?.options?.isHTML5" nowrap>{{ item?.file?.size/1024/1024 | number:'.2' }} MB</td>

            </tr>

            </tbody>

        </table>

        <div>

            <div>

                Queue progress:

                <div class="progress">

                    <div class="progress-bar" role="progressbar" [ngStyle]="{ 'width': uploader?.progress + '%' }"></div>

                </div>

            </div>

            <button type="button" class="btn btn-success btn-s"

                    (click)="uploader?.uploadAll()" [disabled]="!uploader?.getNotUploadedItems()?.length">

                <span class="fa fa-upload"></span> Upload all

            </button>

            <button type="button" class="btn btn-warning btn-s"

                    (click)="uploader?.cancelAll()" [disabled]="!uploader?.isUploading">

                <span class="fa fa-ban"></span> Cancel all

            </button>

            <button type="button" class="btn btn-danger btn-s"

                    (click)="uploader?.clearQueue()" [disabled]="!uploader?.queue?.length">

                <span class="fa fa-trash"></span> Remove all

            </button>

        </div>

    </div>

</div>

<!--photo uploader ends-->