//IMAGE DROPZONE  
***Dropzone***.autoDiscover = false  
const ***myDropzone*** = new ***Dropzone***('#my-dropzone', {  
 type: 'GET',  
 url: 'upload/',  
 init: function () {  
 this.on('sending', function (file, xhr, formData) {  
 formData.append('csrfmiddlewaretoken', ***csrftoken***)  
 formData.append('file-name', file.name)  
 formData.append('file-type', file.type)  
 formData.append('event\_id', ***eventId***)  
 formData.append('user', ***eventUser***)  
 ***console***.log(file, xhr, formData)  
  
 process(file)  
  
  
 })  
 },  
 resizeWidth: 600,  
 resizeHeight: 600,  
 maxFiles: 10,  
 maxFilesize: 10,  
 resizeQuality: .7,  
 acceptedFiles: '.png, .jpg, .jpeg',  
})  
  
  
//VIDEO DROPZONE  
***Dropzone***.autoDiscover = false  
const ***myVideoDropzone*** = new ***Dropzone***('#my-video-dropzone', {  
 type: 'GET',  
 url: 'upload\_video/',  
 init: function () {  
 this.on('sending', function (file, xhr, formData) {  
 formData.append('csrfmiddlewaretoken', ***csrftoken***)  
 formData.append('file-name', file.name)  
 formData.append('file-type', file.type)  
 formData.append('event\_id', ***eventId***)  
 formData.append('user', ***eventUser***)  
 ***console***.log(file, xhr, formData)  
 getVideoSignedRequest(file)  
 })  
 },  
 maxFiles: 1,  
 maxFilesize: 50,  
 resizeWidth: 600,  
 resizeHeight: 600,  
 resizeQuality: .7,  
 acceptedFiles: '.mp4',  
  
})  
  
  
//Get the CSRF  
const getCookie = (name) => {  
 let cookieValue = null;  
 if (***document***.cookie && ***document***.cookie !== '') {  
 const cookies = ***document***.cookie.split(';');  
 for (let i = 0; i < cookies.length; i++) {  
 const cookie = cookies[i].trim();  
 // Does this cookie string begin with the name we want?  
 if (cookie.substring(0, name.length + 1) === (name + '=')) {  
 cookieValue = decodeURIComponent(cookie.substring(name.length + 1));  
 break;  
 }  
 }  
 }  
 return cookieValue;  
}  
const ***csrftoken*** = getCookie('csrftoken')

//MY COMPRESSION ATTEMPT

function process(file) {

if (!file) return;

const reader = new ***FileReader***();

reader.readAsDataURL(file);

reader.onload = function (event) {

const imgElement = ***document***.createElement("img");

imgElement.src = event.target.result;

imgElement.onload = function (e) {

const canvas = ***document***.createElement("canvas");

const MAX\_WIDTH = 400;

const scaleSize = MAX\_WIDTH / e.target.width;

canvas.width = MAX\_WIDTH;

canvas.height = e.target.height \* scaleSize;

const ctx = canvas.getContext("2d");

ctx.drawImage(e.target, 0, 0, canvas.width, canvas.height);

const srcEncoded = ctx.canvas.toDataURL(e.target, .8);

// you can send srcEncoded to the server

getSignedRequest(srcEncoded);

};

};

}

//IMAGES: DIRECT UPLOAD FOR IMAGES PLACED IN DROPZONE

function getSignedRequest(file){

***console***.log('starting getSignedRequest for ' + file.name);

const xhr = new ***XMLHttpRequest***();

***console***.log('making GET');

xhr.open('GET', `/upload?file-name=${file.name}&file-type=${file.type}&event\_id=${***eventId***}`);

***console***.log('made GET');

xhr.onreadystatechange = () => {

if(xhr.readyState === 4){

if(xhr.status === 200){

***console***.log('xhr.status is 200 for the signed request');

const response = ***JSON***.parse(xhr.responseText);

uploadFile(file, response.data, response.url);

}

else{

alert('Could not get signed URL.');

***console***.log('could not get signed url');

}

}

};

xhr.send();

}

function uploadFile(file, s3Data, url){

***console***.log('starting upload for ' + file.name);

const xhr = new ***XMLHttpRequest***();

xhr.upload.addEventListener("progress", updateProgress);

xhr.open('POST', s3Data.url);

xhr.setRequestHeader('x-amz-acl', 'public-read');

const postData = new ***FormData***();

for(key in s3Data.fields){

postData.append(key, s3Data.fields[key]);

}

postData.append('file', file);

xhr.onreadystatechange = () => {

if(xhr.readyState === 4){

if(xhr.status === 200 || xhr.status === 204){

$('#success').removeClass('hidden');

var num = $('#success').attr('data-num');

num ++;

$('#success').attr('data-num', num);

***console***.log(file.name + ' File(s) Successfully Uploaded')

$('#success-message').html(num + ' Successfully Uploaded');

$('#success-list').append('<li>' + file.name + '</li>');

}

else{

$('#failure').removeClass('hidden');

var num = $('#failure').attr('data-num');

num ++;

$('#failure').attr('data-num', num);

$('#failures-message').html(num + ' File(s) Not Uploaded');

$('#failures-list').append('<li>' + file.name + '</li>');

}

}

};

xhr.send(postData);

}

function updateProgress(oEvent) {

***console***.log('updateProgress running...');

$('.progress').removeClass('hidden');

if (oEvent.lengthComputable) {

var percentComplete = oEvent.loaded / oEvent.total;

$('.progress-bar').css('width', (percentComplete)\*100+'%').text(***Math***.floor((percentComplete)\*100));

} else {

$('.progress-bar').html('Unable to calculate file size... file is uploading...')

}

}