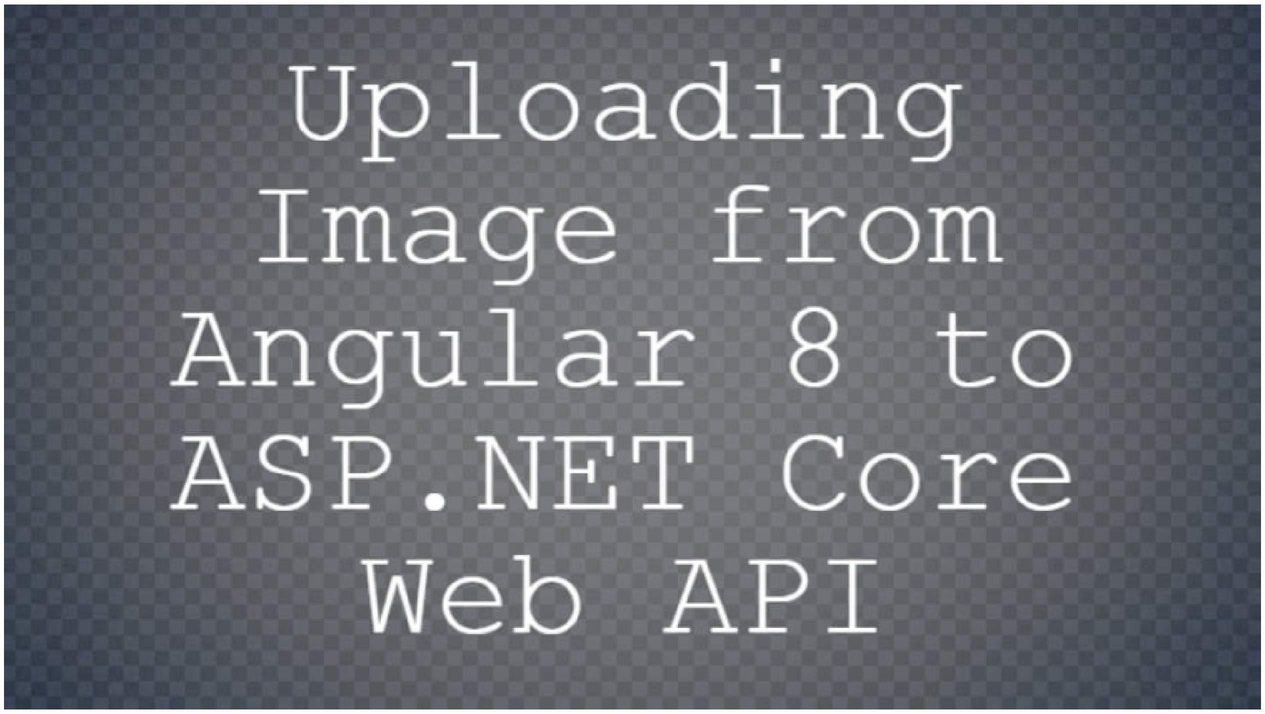
Uploading Image from Angular to ASP.NET Core Web API

Hemant Joshi Jul 19 '19 *Originally published at* codingdefined.com *on Jul 19, 2019*
· 3 min read

#angular **#aspnetcore**



In this post we will be discussing about creating an application where you can upload image from angular 8 to ASP.NET Core Web API using ASP.NET Boilerplate. In this we will be going through Back End first i.e. ASP.NET Core part and then we will go through the frontend i.e. Angular.

I have written a function UploadProfilePicture in UserAppService which will handle the upload of profile pic and then create image on a predefined location.

The logic behind it very straight forward where we are providing the path where the file will be stored. We check if the directory exists or not, if not then creating the directory. Then we are creating the file stream object and then storing the file in that location.

```
public async Task<string> UploadProfilePicture([FromForm(Name = "uploa
{
  if (file == null || file.Length == 0)
    throw new UserFriendlyException("Please select profile picture");

  var folderName = Path.Combine("Resources", "ProfilePics");
  var filePath = Path.Combine(Directory.GetCurrentDirectory(), folderN

  if (!Directory.Exists(filePath))
  {
    Directory.CreateDirectory(filePath);
  }

  var uniqueFileName = $"{userId}_profilepic.png";
  var dbPath = Path.Combine(folderName, uniqueFileName);

  using (var fileStream = new FileStream(Path.Combine(filePath, unique
  {
    await file.CopyToAsync(fileStream);
  }

  return dbPath;
}
```

Few things to note here :

1. IFormFile object will come as null if the name attribute of the file input is not same as that of the name of the parameter used in the controller. So you should name then same or use [FromForm(Name = "")] and then assign the name as shown below.

2. Since we are creating a new directory we need to tell ASP.NET Core to serve the static files from that location. For that we need to modify the Configure method of Startup.cs class as shown below :

```
app.UseStaticFiles(new StaticFileOptions()
{
 FileProvider = new PhysicalFileProvider(Path.Combine(Directory.GetCur
 RequestPath = new PathString("/Resources")
});
```

1. When using IFormFile, the swagger will give you multiple text boxes like ContentType, ContentDisposition, Headers, Length, Name, FileName etc instead of file upload control. To change the textboxes to the actual file upload control we need to implement IOperationFilter and then implement the apply method as shown below. The main part is type where we need to define file, since we are clearing all the previous parameters we also need to add the user id parameter.

```
using Swashbuckle.AspNetCore.Swagger;
using Swashbuckle.AspNetCore.SwaggerGen;

namespace LetsDisc.Web.Host.Startup
{
 public class FileUploadOperation : IOperationFilter
 {
  public void Apply(Operation operation, OperationFilterContext contex
  {
   if (operation.OperationId.ToLower().Contains("upload"))
   {
    operation.Parameters.Clear();
    operation.Parameters.Add(new NonBodyParameter
        {
          Name = "uploadedFile",
          In = "formData",
          Description = "Upload File",
          Required = true,
          Type = "file"
        });
    operation.Parameters.Add(new NonBodyParameter
        {
          Name = "userId",
          In = "query",
          Description = "",
          Required = true,
          Type = "long"
        });
    operation.Consumes.Add("multipart/form-data");
   }
  }
 }
}
```

Thus we are finished with the backend, now we will go forward with the frontend implementation.

HTML

In HTML we will have an input of type file and then we have both change and click function so that user can upload the same image twice.

```
<div class="col-md-3 profile-image-edit">
 <label class="hoverable" for="fileInput">
 ... img tag
  <span class="hover-text">Choose file</span>
  <span class="background"></span>
 </label>
 <br />
 <input #fileInput id="fileInput" type='file' (click)="fileInput.value
 <button class="btn btn-default" *ngIf="url" (click)="delete()">delete
</div>
```

CSS

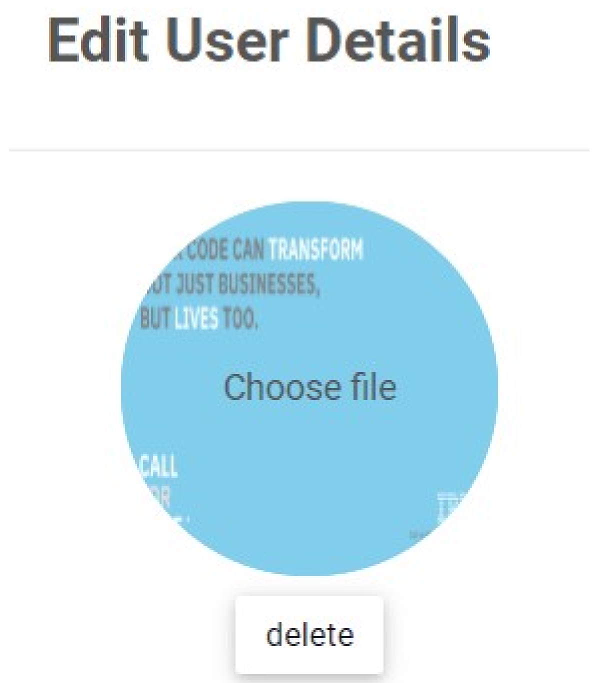
```
.hoverable {
    position: relative;
    cursor: pointer;
    height: 150px;
    width: 150px;
    border-radius: 50%;
}

    .hoverable .hover-text {
        position: absolute;
        display: none;
        top: 50%;
        left: 50%;
        transform: translate(-50%,-50%);
        z-index: 2;
    }

    .hoverable .background {
        position: absolute;
        display: none;
        top: 0;
        left: 0;
        bottom: 0;
        right: 0;
        background-color: rgba(255, 255, 255, 0.5);
        pointer-events: none;
        border-radius: 50%;
        z-index: 1;
    }

    .hoverable:hover .hover-text {
        display: block;
    }

    .hoverable:hover .background {
        display: block;
    }
```

Code Behind

In the code behind we will have a function which has a fileReader ojject to preview the image as well as we will be calling our backend service for uploading the image.

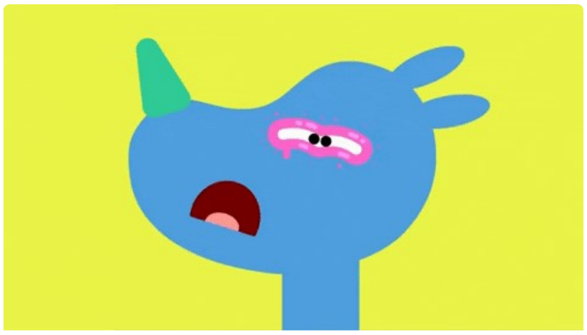
```
onSelectFile(files: FileList) {

  if (files.length === 0)
```

```
      return;

    this.fileToUpload = files.item(0);


    const fileReader: FileReader = new FileReader();
    fileReader.readAsDataURL(this.fileToUpload);

    fileReader.onload = (event: any) => {
      this.url = event.target.result;
    };

    this.files.push({ data: this.fileToUpload, fileName: this.fileToUplc

    this._userService.uploadProfilePicture(this.files[0], this.user.id)
      .subscribe((result: string) => {
        this.userDetails.profileImageUrl = result;
    });
  }

  delete() {
    this.url = null;
  }
```



GitHub Commit: https://github.com/codingdefined/LetsDisc/commit/ab7af63ba3cf94c23278fc2fe00d3769672bf506

Also Published: CodingDefined.com



Sore eyes?

dev.to now has dark mode.

Select **night theme** in the "misc" section of your settings ❤️

Hemant Joshi + FOLLOW
Passionate about FrontEnd Development
@codingdefined codingdefined codingdefined.com

Add to the discussion

PREVIEW SUBMIT

code of conduct - report abuse

Classic DEV Post from May 15 '19

What are your five most used terminal commands?

riscie

community post to share our most used terminal commands.

147 96

Another Post You Might Like

Building scalable robusts and type safe forms with Angular

Maxime

164 19

Another Post You Might Like

Preload All Angular Bundles

John Papa

Learn how to use the built-in Angular preload strategy to preload all lazy loaded bundles in advance

79 6

Let's check the results: how did the Angular perform in 2019?
Radvila1363 - Jan 8

How to Link Your Angular App to Bootstrap via BootstrapCDN
Joseph Adediji - Jan 7

Is 2021 the end of AngularJS?
Chirag Dave - Jan 7

Eu e o Angular
Henrique Lobo (Kico) - Jan 6
