FPT Education

FPT POLYTECHNIC

Lập Trình C# 6

Bài 6: Web API Upload - Blazor

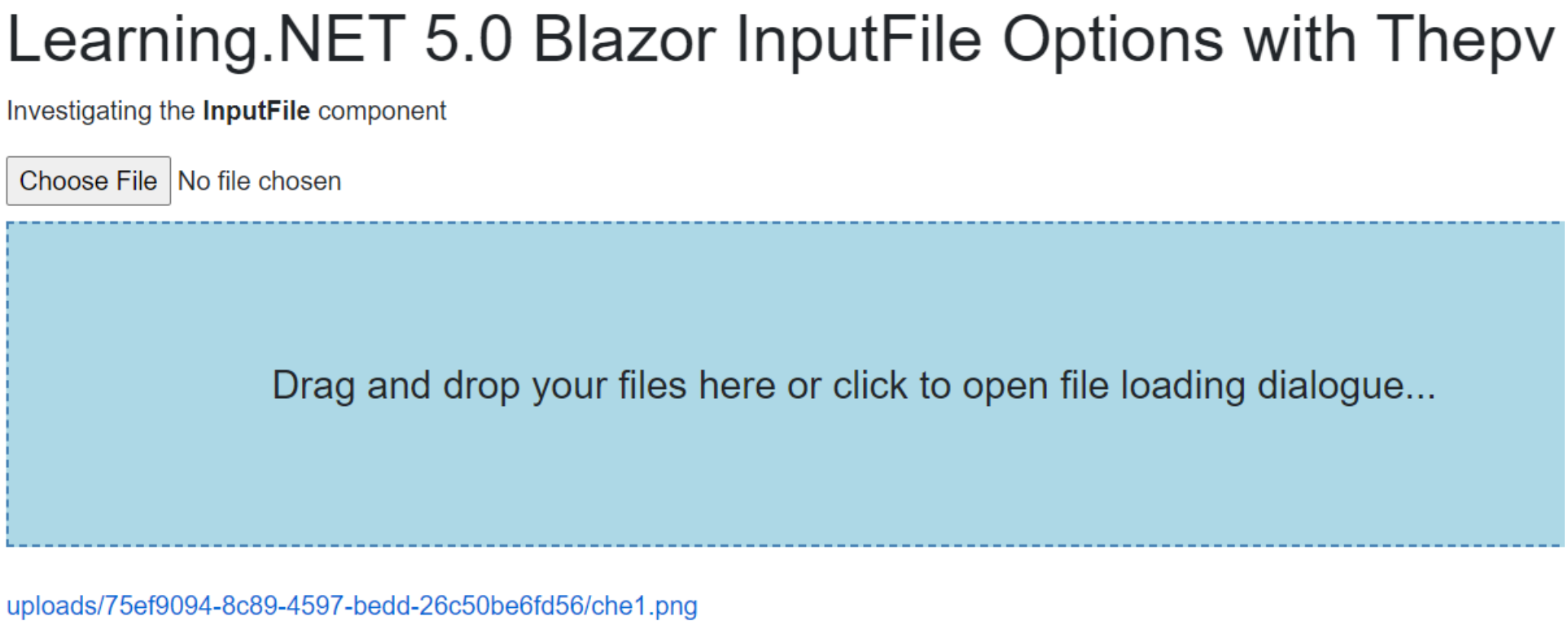
P6.1

Mục tiêu

- Blazor - UPLOAD FILES Drag-drop
- Blazor - UPLOAD FILES API
- Blazor - UPLOAD FILES-DB

BLAZOR - UPLOAD FILES DRAG-DROP

- InputFile component
- ondragenter and ondragleave events



BLAZOR - UPLOAD FILES DRAG-DROP

```
<div>
    <div class="inputArea">
        <InputFile id="inputDefault"
                   OnChange="OnInputFileChange"
                   accept="image/png,image/gif,image/jpeg"/>
    </div>
    <div class="dropArea @dropClass">
        Drag and drop your files here or click to open file loading dialogue.
        <InputFile id="inputDrop"
                   OnChange="OnInputFileChange"
                   @ondragenter="HandleDragEnter"
                   @ondragleave="HandleDragLeave"
                   multiple />
    </div>
```

```
@if (files != null && files.Count > 1)
{
    <div>
        <ul>
            @foreach (var file in files)
            {
                <li>@file.Name</li>
            }
        </ul>
    </div>
}
@if (urls.Count > 0)
{
    foreach (var url in urls)
    {
        <br />
        <a href="@url" download>@url</a>
    }
}
```

BLAZOR - UPLOAD FILES DRAG-DROP

❑ @code section

```
@code {
    IReadOnlyList<IBrowserFile> files;
    List<string> urls = new List<string>();
    string dropClass = string.Empty;
    const int maxFileSize = 10485760;

    private void HandleDragEnter()
    {
        dropClass = "dropAreaDrug";
    }

    private void HandleDragLeave()
    {
        dropClass = string.Empty;
    }
```

BLAZOR - UPLOAD FILES DRAG-DROP

❑ @code section

```
private async Task<string> SaveFile(IBrowserFile file, string guid = null)
{
    if (guid == null)
    {
        guid = Guid.NewGuid().ToString();
    }

    var relativePath = Path.Combine("uploads", guid);
    var dirToSave = Path.Combine(_env.WebRootPath, relativePath);
    var di = new DirectoryInfo(dirToSave);
    if (!di.Exists)
    {
        di.Create();
    }

    var filePath = Path.Combine(dirToSave, file.Name);
    using (var stream = file.OpenReadStream(maxFileSize))
    {
        using (var mstream = new MemoryStream())
        {
            await stream.CopyToAsync(mstream);
            await File.WriteAllBytesAsync(filePath, mstream.ToArray());
        }
    }

    var url = Path.Combine(relativePath, file.Name).Replace("\\", "/");
    return url;
}
```

BLAZOR - UPLOAD FILES DRAG-DROP

- @code section

```
private async Task<List<string>> SaveFiles(IReadOnlyList<IBrowserFile> files)
{
    var list = new List<string>();
    var guid = Guid.NewGuid().ToString();
    foreach (var file in files)
    {
        var url = await SaveFile(file, guid);
        list.Add(url);
    }
    return list;
}
```

BLAZOR - UPLOAD FILES DRAG-DROP

- @code section

```
async Task OnInputFileChange(InputFileChangeEventArgs e)
{
    dropClass = string.Empty;
    try
    {
        if (e.FileCount > 1)
        {
            files = e.GetMultipleFiles();
            urls.Clear();
            urls.AddRange(await SaveFiles(files));
        }
        else
        {
            files = null;

            var url = await SaveFile(e.File);
            urls.Clear();
            urls.Add(url);
        }
    }
    catch (Exception ex)
    {
        System.Diagnostics.Debug.WriteLine(ex.Message);
        throw;
    }
```

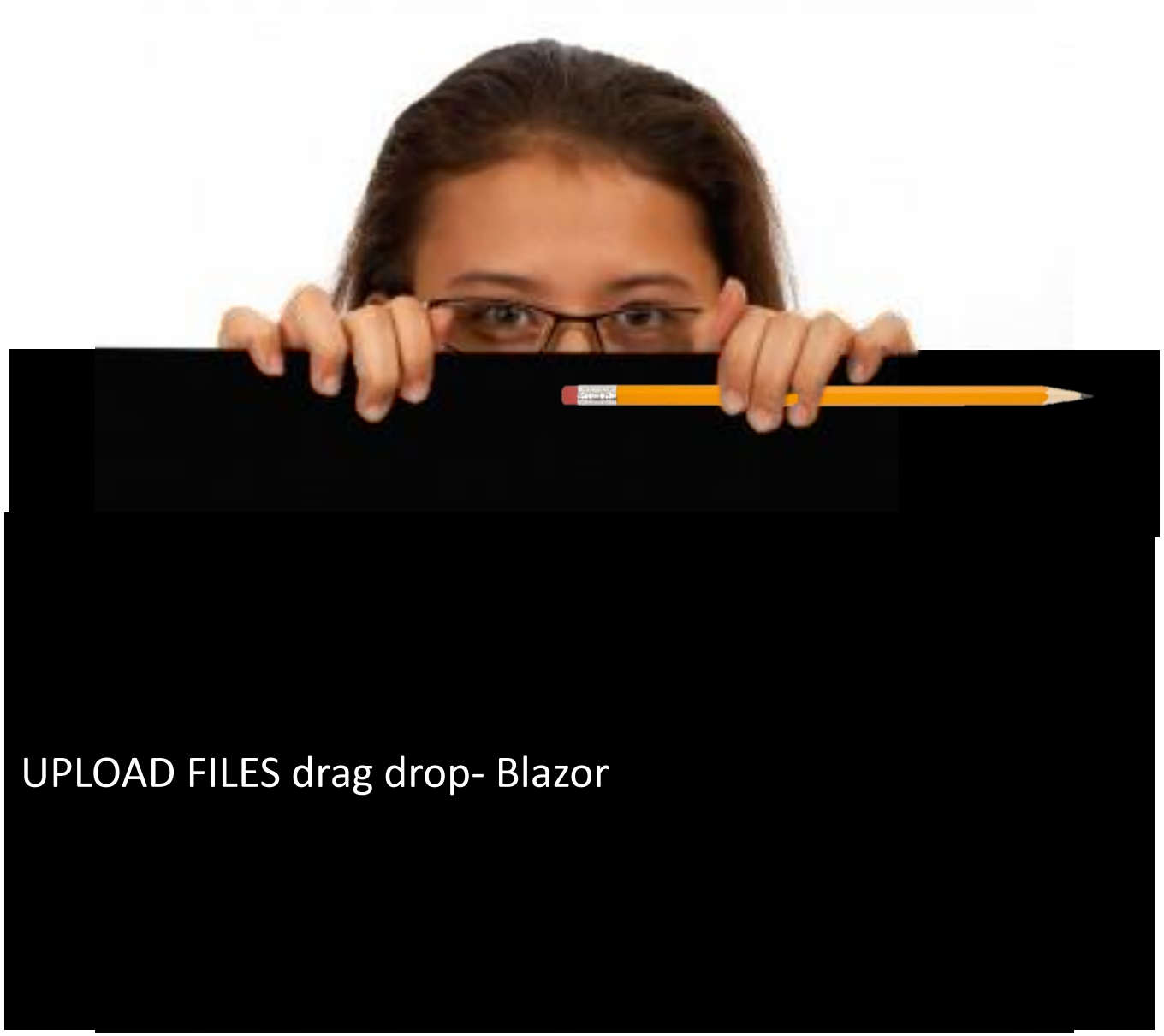
BLAZOR - UPLOAD FILES DRAG-DROP

Styles Css

```
.dropArea {
    border: 2px dashed steelblue;
    padding: 10px;
    display: flex;
    align-items: center;
    justify-content: center;
    background-color: lightblue;
    font-size: 1.5rem;
    cursor: pointer;
    position: relative;
    min-height:200px;
}

.dropArea:hover {
    background-color: lightskyblue;
    color: #333;
}
```

```
.dropArea input[type=file] {
    position: absolute;
    width: 100%;
    height: 100%;
    opacity: 0;
    cursor: pointer;
}

.dropAreaDrug {
    background-color: lightseagreen;
}
```

DEMO 6.1



FPT Education

FPT POLYTECHNIC

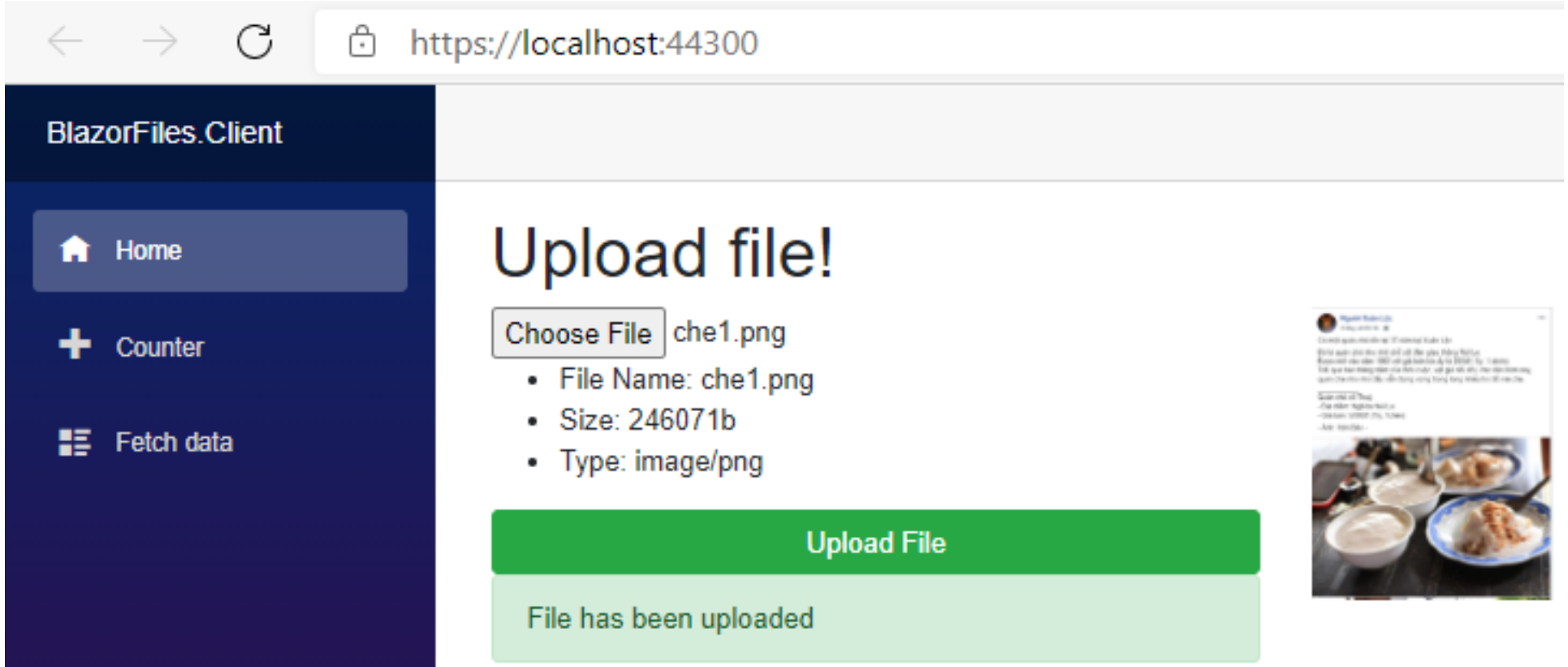
LẬP TRÌNH C# 6

BÀI 6: WEB API UPLOAD - BLAZOR

P6.2

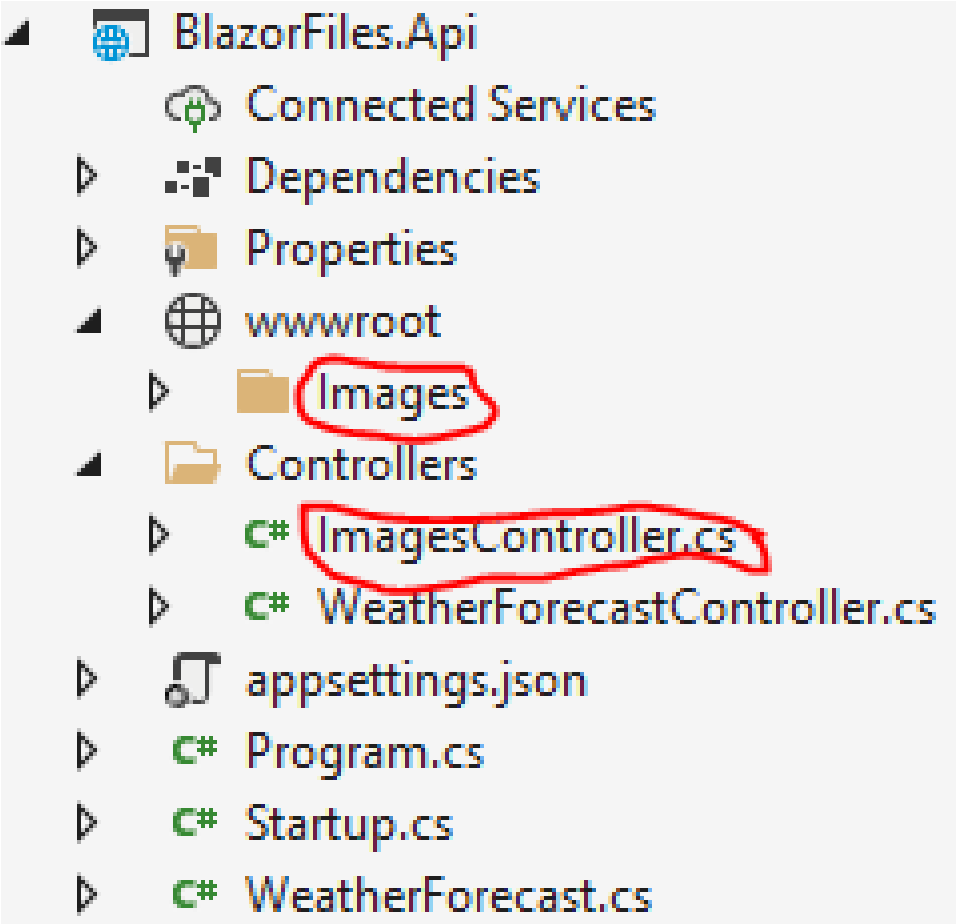
BLAZOR - UPLOAD FILES

- File Upload – ASP.NET Core Web API Implementation
- File Upload with Blazor WebAssembly



BLAZOR - UPLOAD FILES

- BlazorFilesApi

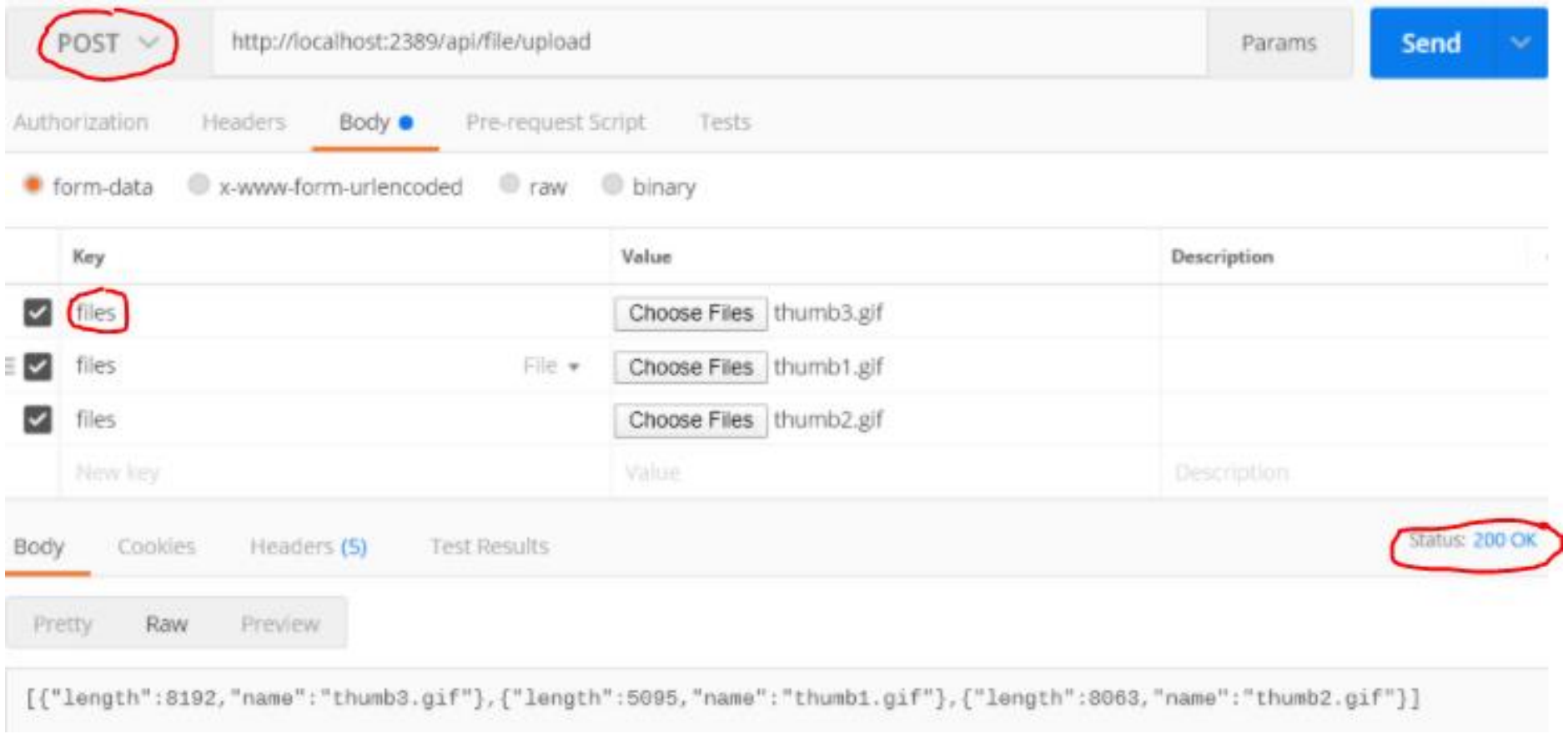


Controller(ImagesController)

```
private readonly IHostEnvironment _environment;
0 references | 0 exceptions
public ImagesController(IHostEnvironment environment)
{
    this._environment = environment;
}
[HttpPost]
0 references | 0 requests | 0 exceptions
public async Task<IActionResult> Post([FromForm]IFormFile image)
{
    if (image == null || image.Length == 0)
        return BadRequest("Upload a file");

    string fileName = image.FileName;
    string extension = Path.GetExtension(fileName);

    string[] allowedExtensions = { ".jpg", ".png", ".bmp" };

    if (!allowedExtensions.Contains(extension))
        return BadRequest("File is not a valid image");

    string newFileName = $"{Guid.NewGuid()}{extension}";
    string filePath = Path.Combine(_environment.ContentRootPath, "wwwroot", "Images", newFileName);

    using (var fileStream = new FileStream(filePath, FileMode.Create, FileAccess.Write))
    {
        await image.CopyToAsync(fileStream);
    }
    return Ok($"Images/{newFileName}");
```

BLAZOR - UPLOAD FILES

Test api



BlazorFilesClient

Tewr.Blazor.FileReader by Tor Knutsson (Tewr)

Create Read-Only file streams from file input elements or drop targets in Blazor.

```
@functions
{
    ElementReference inputReference;
    string message = string.Empty;
    string imagePath = null;//get url image
    string fileName = string.Empty;
    string type = string.Empty;
    string size = string.Empty;
    Stream fileStream = null;
    async Task OpenFileAsync()
    {
        // Read the files
        var file = (await fileReader.CreateReference(inputReference).EnumerateFilesAsync())
            .FirstOrDefault();
        if (file == null)
            return;
        // Get the info of that files
        var fileInfo = await file.ReadFileInfoAsync();
        fileName = fileInfo.Name;
        size = $"{fileInfo.Size}b";
        type = fileInfo.Type;
        using (var ms = await file.CreateMemoryStreamAsync((int)fileInfo.Size))
        {
            fileStream = new MemoryStream(ms.ToArray());
        }
    }
```

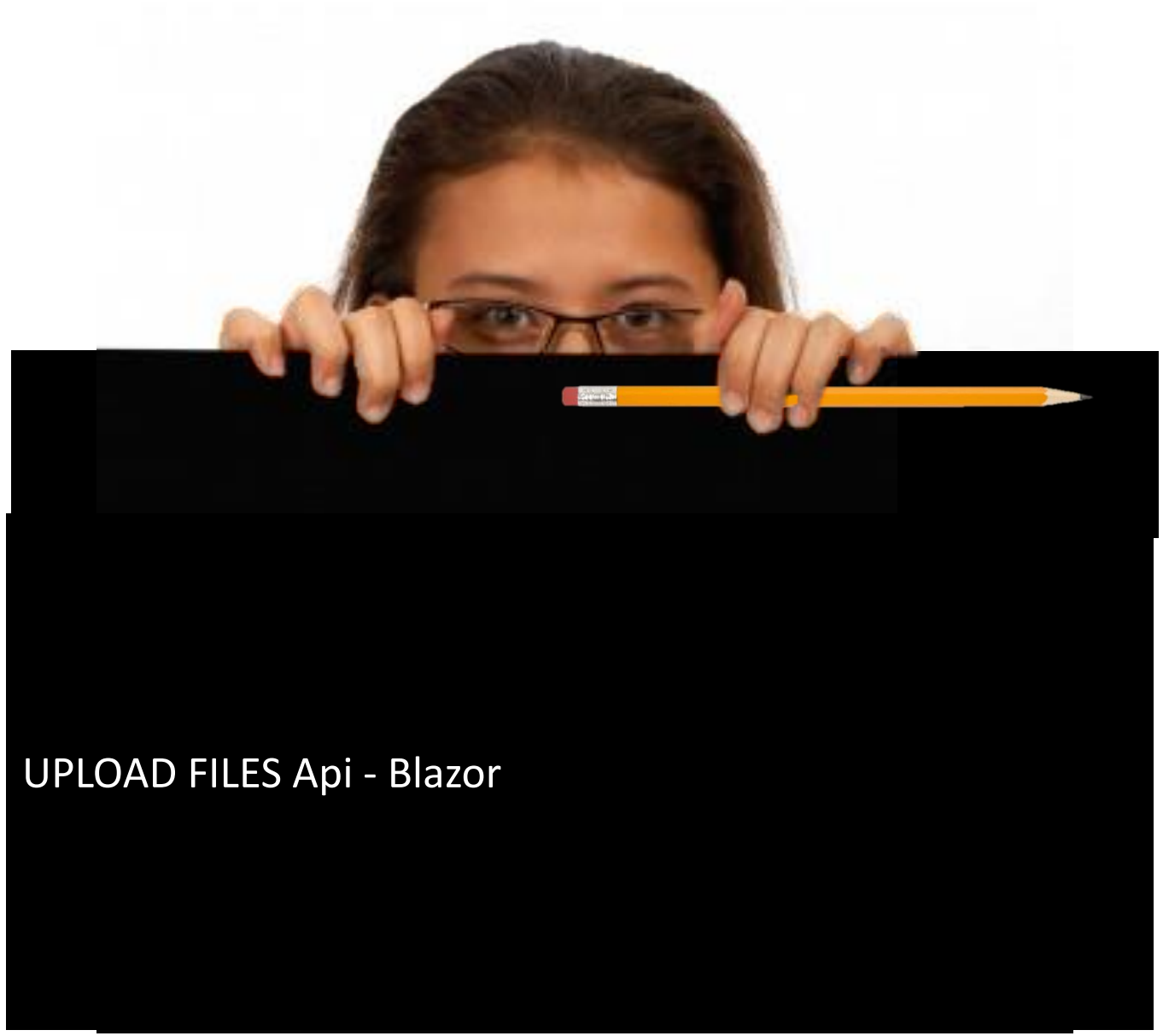
BlazorFilesClient

```
async Task UploadFileAsync()
{
    // Create the content
    var content = new MultipartFormDataContent();
    content.Headers.ContentDisposition = new System.Net.Http.Headers.ContentDispositionHeaderValue
        ("form-data");
    content.Add(new StreamContent(fileStream, (int)fileStream.Length), "image", fileName);
    string url = "https://localhost:44345";
    var response = await client.PostAsync($"{url}/api/images", content);
    if(response.IsSuccessStatusCode)
    {
        var uploadedFileName = await response.Content.ReadAsStringAsync();
        imagePath = $"{url}/{uploadedFileName}";
        message = "File has been uploaded successfully!";
    }
}
```

BlazorFilesClient

```
@inject IFileReaderService fileReader
@inject HttpClient  client
<h1>Upload file!</h1>
<div class="row">
    <div class="col-4">
        <div class="form-group">
            <input type="file" @ref="inputReference" @onchange="async () => await OpenFileAsync()"/>
            <ul>
                <li>File Name: @fileName</li>
                <li>Size: @size</li>
                <li>Type: @type</li>
            </ul>
        </div>
        <button class="btn btn-block btn-success" @onclick="async () => await UploadFileAsync()">Upload File

        @if (!string.IsNullOrWhiteSpace(message))
        {
            <div class="alert alert-success">
                File has been uploaded
            </div>
        }
    </div>
    <div class="col-4">
        @if (imagePath != null)
        {
            <img style="width:150px" src="@imagePath" />
        }
```

DEMO 6.2



FPT Education

FPT POLYTECHNIC

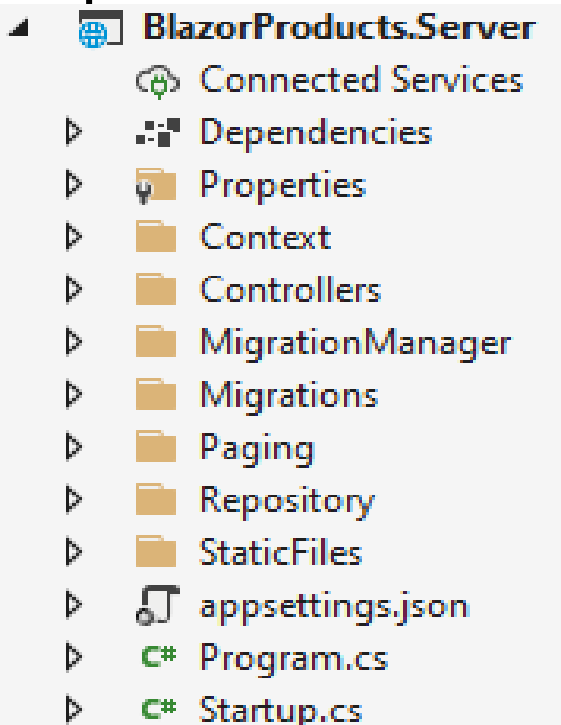
LẬP TRÌNH C# 6

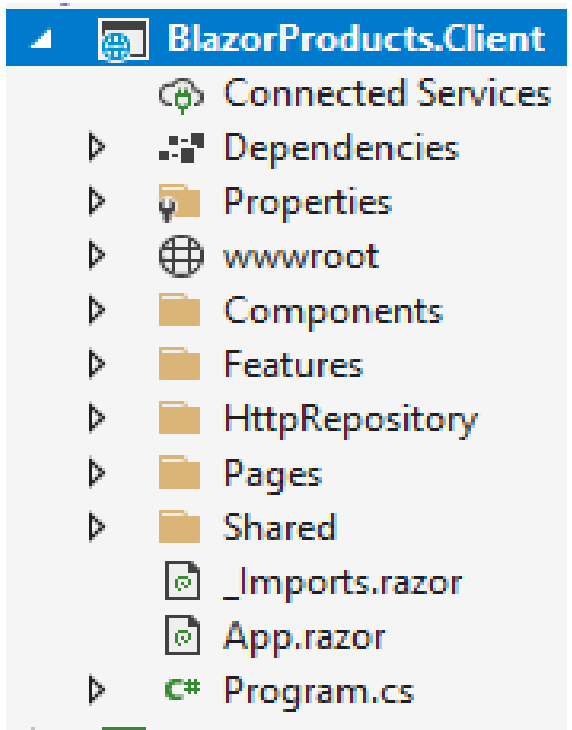
BÀI 6: WEB API UPLOAD - BLAZOR

P6.3

BLAZOR - UPLOAD FILES-DB

- Code first
- Api upload
  - Save file vào thư mục
  - Save url file xuống db
- File Upload with Blazor WebAssembly





BLAZOR - UPLOAD FILES-DB

- Code first

```
public class Product
{
    10 references
    public Guid Id { get; set; }
    [Required(ErrorMessage = "Name is required field")]
    13 references
    public string Name { get; set; }
    [Required(ErrorMessage = "Supplier is required field")]
    10 references
    public string Supplier { get; set; }
    [Range(1, double.MaxValue, ErrorMessage = "Value for the
    10 references
    public double Price { get; set; }
    10 references
    public string ImageUrl { get; set; }
}
```

```
public DbSet<Product> Products { get; set; }
```

```
"ConnectionStrings": {
  "sqlConnection": "server=.; database=UploadBlazor; Integrated Security=true"
},
```

```
services.AddDbContext<ProductContext>(opt => opt.UseSqlServer
(Configuration.GetConnectionString("sqlConnection")));
```

BLAZOR - UPLOAD FILES-DB

- Api upload
  - Show product
  - Create product

```
public async Task CreateProduct(Product product)
{
    _context.Add(product);
    await _context.SaveChangesAsync();
}
```

```
[HttpPost]
0 references
public async Task<IActionResult> CreateProduct([FromBody] Product product)
{
    if (product == null)
        return BadRequest();
    await _repo.CreateProduct(product);
    return Created("", product);
}
```

BLAZOR - UPLOAD FILES-DB

Upload file to folder

```
[HttpPost]
0 references
public IActionResult Upload()
{
    try
    {
        var file = Request.Form.Files[0];
        var folderName = Path.Combine("StaticFiles", "Images");
        var pathToSave = Path.Combine(Directory.GetCurrentDirectory(), folderName);

        if (file.Length > 0)
        {
            var fileName = ContentDispositionHeaderValue.Parse(file.ContentDisposition).FileName.Trim('"');
            var fullPath = Path.Combine(pathToSave, fileName);
            var dbPath = Path.Combine(folderName, fileName);

            using (var stream = new FileStream(fullPath, FileMode.Create))
            {
                file.CopyTo(stream);
            }

            return Ok(dbPath);
        }
        else
        {
            return BadRequest();
        }
    }
    catch (Exception ex)
```

BLAZOR - UPLOAD FILES-DB

BlazorClient

```
public async Task CreateProduct(Product product)
{
    var content = JsonSerializer.Serialize(product);
    var bodyContent = new StringContent(content, Encoding.UTF8, "application/json");

    var postResult = await _client.PostAsync("https://localhost:5011/api/products", bodyContent);
    var postContent = await postResult.Content.ReadAsStringAsync();

    if(!postResult.IsSuccessStatusCode)
    {
        throw new ApplicationException(postContent);
    }
}
public async Task<string> UploadProductImage(MultipartFormDataContent content)
{
    var postResult = await _client.PostAsync("https://localhost:5011/api/upload", content);
    var postContent = await postResult.Content.ReadAsStringAsync();
    if (!postResult.IsSuccessStatusCode)
    {
        throw new ApplicationException(postContent);
    }
    else
    {
        var imgUrl = Path.Combine("https://localhost:5011/", postContent);
        return imgUrl;
    }
```

BLAZOR - UPLOAD FILES-DB

❑ BlazorClient

```
<EditForm Model="_product" OnValidSubmit="Create" class="card card-body bg-light r
    <DataAnnotationsValidator />
    <div class="form-group row">
        <label for="name" class="col-md-2 col-form-label">Name:</label>
        <div class="col-md-10">
            <InputText id="name" class="form-control" @bind-Value="_product.Name"
            <ValidationMessage For="@(() => _product.Name)" />
        </div>
    </div>
    .............
```

```
<div class="form-group row">
    <label for="image" class="col-md-2 col-form-label">Image:</label>
    <div class="col-md-10">
        <ImageUpload OnChange="AssignImageUrl" />
    </div>
</div>

<div class="row">
    <div class="col-md-12 text-right">
        <button type="submit" class="btn btn-success">Create</button>
    </div>
```

BLAZOR - UPLOAD FILES-DB

❑ BlazorClient

```
public partial class CreateProduct
{
    private Product _product = new Product();
    private SuccessNotification _notification;
    [Inject]
    1 reference | 0 exceptions
    public IProductHttpRepository ProductRepo { get; set; }
    1 reference | 0 exceptions
    private async Task Create()
    {
        await ProductRepo.CreateProduct(_product);
        _notification.Show();
    }
    1 reference | 0 exceptions
    private void AssignImageUrl(string imgUrl) => _product.ImageUrl = imgUrl;
}
```

BLAZOR - UPLOAD FILES-DB

❑ BlazorClient upload

```
<input type="file" @ref="_input" @onchange="HandleSelected" accept=".jpg, .jpeg, png" />
@if (ImgUrl != null)
{
<div>
    <img src="@ImgUrl" style="width:300px" />
</div>
```

BLAZOR - UPLOAD FILES-DB

❑ BlazorClient upload

```
private ElementReference _input;
[Parameter]
4 references | 0 exceptions
public string ImgUrl { get; set; }
[Parameter]
1 reference | 0 exceptions
public EventCallback<string> OnChange { get; set; }
[Inject]
1 reference | 0 exceptions
public IFileReaderService FileReaderService { get; set; }
[Inject]
1 reference | 0 exceptions
public IProductHttpRepository Repository { get; set; }
1 reference | 0 exceptions
private async Task HandleSelected()
{
    foreach (var file in await FileReaderService.CreateReference(_input).EnumerateFilesAsync())
    {
        if (file != null)
        {
            var fileInfo = await file.ReadFileInfoAsync();
            using (var ms = await file.CreateMemoryStreamAsync(4 * 1024))
            {
                var content = new MultipartFormDataContent();
                content.Headers.ContentDisposition = new ContentDispositionHeaderValue("form-data");
                content.Add(new StreamContent(ms, Convert.ToInt32(ms.Length)), "image", fileInfo.Name);

                ImgUrl = await Repository.UploadProductImage(content);

                await OnChange.InvokeAsync(ImgUrl);
```

DEMO 6.3

UPLOAD File database blazor

Tổng kết bài học

- Blazor - UPLOAD FILES Drag-drop
- Blazor - UPLOAD FILES API
- Blazor - UPLOAD FILES-DB

KẾT THÚC