Modul 07: ASP.NET Core – File Upload

Ziel:	*Integration eines Bild-Uploads in die MovieStore-App unter Verwendung des bestehenden FileServers.*
Tools:	*Visual Studio 2022*
Dauer:	*~30min*

1 Movie-Modell erweitern

Ergänzen Sie jeweils das Domänen Modell und das ViewModel um eine weitere Eigenschaft **ImagePath** bzw. **Image**.

```
public IFormFile? Image { get; set; }
```

Beachten Sie, dass nur die Web-UI und damit auch nur das ViewModel den Typ `IFormFile` kennt.

2 Ansichten anpassen

Ergänzen Sie in der **Create.cshtml** ein Dateiupload-Feld und achten Sie darauf den MIME-Typ über das Encoding Attribut zu setzen.

```
<form method="post" asp-action="Create" enctype="multipart/form-data">
    <!-- ... -->
    <div class="form-group">
        <label asp-for="Image" class="control-label"></label>
        <input type="file" asp-for="Image" class="form-control" />
        <span asp-validation-for="Image" class="text-danger"></span>
    </div>
    <!-- ... -->
</form>
```

In der **Details.cshtml** soll das Bild ebenfalls angezeigt werden.

```
@if (!string.IsNullOrEmpty(Model.ImagePath))
{
    <img src="@Model.ImagePath"
         class="img-fluid"
         alt="Filmplakat"
         style="max-height: 300px;">
}
```

3 FileService implementieren

Implementieren Sie den FileService. Er könnte ungefähr so aussehen:

```
public record FileServiceOptions(string BaseUrl, string ApiKey);

public class RemoteFileService : IFileService
{
    private readonly HttpClient _httpClient;

    public RemoteFileService(IOptions<FileServiceOptions> options,
                             HttpClient httpClient)
    {

        _httpClient = httpClient;
        _httpClient.BaseAddress = new Uri(options.Value.BaseUrl);
        _httpClient.DefaultRequestHeaders.Add("X-API-Key", options.Value.ApiKey);
    }

    public async Task<string> UploadFile(string fileName, Stream stream)
    {
        var content = new StreamContent(stream);
        content.Headers.ContentType = new MediaTypeHeaderValue(
               MediaTypeNames.Application.Octet
        );

        using var formContent = new MultipartFormDataContent();
        formContent.Add(content, "file", fileName);

        var response = await _httpClient.PostAsync("upload", formContent);
        response.EnsureSuccessStatusCode();
        return await response.Content.ReadAsStringAsync();
    }
}
```

Registrieren Sie den HttpClient, IFileService und Konfiguration des FileServers in der **Program.cs**. Die **appsettings.json** können Sie um den folgenden Schritt ergänzen.

```
"FileServer": {
  "BaseUrl": "http://localhost:5101/",
  "ApiKey": "7F12CA09-0EE3-461B-8BA2-3059E3A855CD"
},
```

4 Testen der Funktionalität

Starten Sie die **FileServer.exe** aus dem bin-Verzeichnis oder aus Visual Studio heraus: Rechtsklick auf das Projekt *FileServer > Debug > Start Without Debugging*.

Starten Sie dann die Web-Applikation im Debug Modus und legen Sie einen neuen Film mit Titelbild an. Kontrollieren Sie auch die Detailansicht. Falls das Bild nicht angezeigt wird, prüfen Sie, ob es auf dem FileServer erreichbar ist und Debuggen Sie sich Schritt für Schritt aufmerksam durch die Applikation.