# 1 Endpunkte mit CRUD-Operationen

> **Hinweis:** Dieser Auftrag setzt auf erledigtem Auftrag *02-AA-Minimal-API-MongoDB* auf.

## 1.1 Ziele

- Sie unterscheiden GET, POST, PUT und DELETE-Requests
- Sie erstellen CRUD-Endpunkte für ein Web-API
- Sie rufen ein Web-API mit Postman auf.

## 1.2 Umgebung

Die Übung wird auf der VM LP-22.04 durchgeführt.

## 1.3 Aufgaben

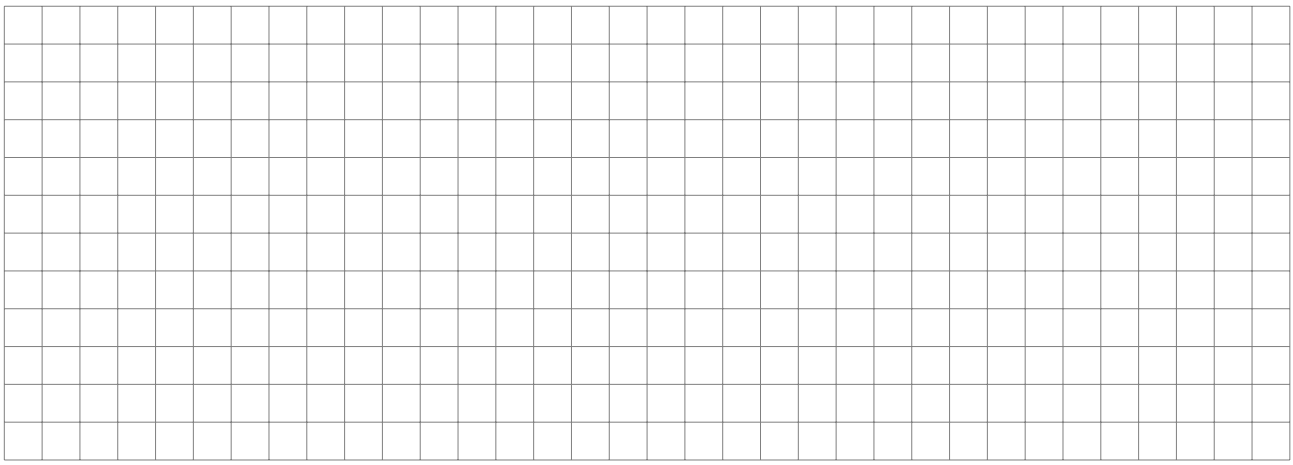
Aufgabe 1: Installation Postman | Einzelarbeit | 10’

Installieren Sie als Client für den Aufruf des WebApis die API-Plattform Postman:

```
snap install postman
```

Aufgabe 2: HTTP-Methoden für CRUD-Operationen | Einzelarbeit | 15’

Entnehmen Sie Artikel Using HTTP Methods for RESTful Services, welche HTTP-Methode für welche CRUD-Operation verwendet wird.



Aufgabe 3: Service-Endpunkte | Einzelarbeit | 35'

Um die verschiedenen Eigenschaften eines Films abzubilden, erstellen Sie unter *min-api-with-mongo/WebApi* ein neues File *Movie.cs* mit folgendem Codegerüst:

```
public class Movie
{
    public string Id { get; set; } = "";

    public string Title { get; set; } = "";
    public int Year { get; set; }
    public string Summary { get; set; } = "";
    public string[] Actors { get; set; } = Array.Empty<string>();
}
```

Erweitern Sie *min-api-with-mongo/WebApi/Program.cs* um folgende Endpunkte:

```
// Insert Movie
// Wenn das übergebene Objekt eingefügt werden konnte,
// wird es mit Statuscode 200 zurückgegeben.
// Bei Fehler wird Statuscode 409 Conflict zurückgegeben.
app.MapPost("/api/movies", () =>
{
    throw new NotImplementedException();
});

// Get all Movies
// Gibt alle vorhandenen Movie-Objekte mit Statuscode 200 OK zurück.
app.MapGet("api/movies", () =>
{
    throw new NotImplementedException();
});

// Get Movie by id
// Gibt das gewünschte Movie-Objekt mit Statuscode 200 OK zurück.
// Bei ungültiger id wird Statuscode 404 not found zurückgegeben.
app.MapGet("api/movies/{id}", (string id) =>
{
    throw new NotImplementedException();
});

// Update Movie
// Gibt das aktualisierte Movie-Objekt zurück.
// Bei ungültiger id wird Statuscode 404 not found zurückgegeben.
```

```
app.MapPut("/api/movies/{id}", (string id, Movie movie) =>
{
    throw new NotImplementedException();
});

// Delete Movie
// Gibt bei erfolgreicher Löschung Statuscode 200 OK zurück.
// Bei ungültiger id wird Statuscode 404 not found zurückgegeben.
app.MapDelete("api/movies/{id}", (string id) =>
{
    throw new NotImplementedException();
});
```

Implementieren Sie die Endpunkte mit Beispielcode (ohne DB-Anbindung) so, dass die übergebenen Parameter entgegengenommen werden und ein korrektes Ergebnis (gemäss Beschreibung) zurück geliefert wird.

> **Hinweis** Geben Sie ein C#-Objekt mit Results.Ok(myObject) zurück, wird es im Response im JSON-Format serialisiert und mit HTTP-Statuscode 200 OK zurückgegeben. Geben Sie Results.NotFound() zurück, wird ein HTTP-Statuscode 404 Not found zurückgegeben.

Beispiel für *Get Movie by id*:

```
app.MapGet("api/movies/{id}", (string id) =>
{
    if(id == "1")
    {
        var myMovie = new Movie()
        {
            Id = "1",
            Title = "Asterix und Obelix",
        };
        return Results.Ok(myMovie);
    }
    else
    {
        return Results.NotFound();
    }
});
```

Starten Sie die Anwendung und testen Sie die verschiedenen Enpunkte mit Postman. Vergewissern Sie sich, dass das API auf einen Request von Postman korrekt antwortet.

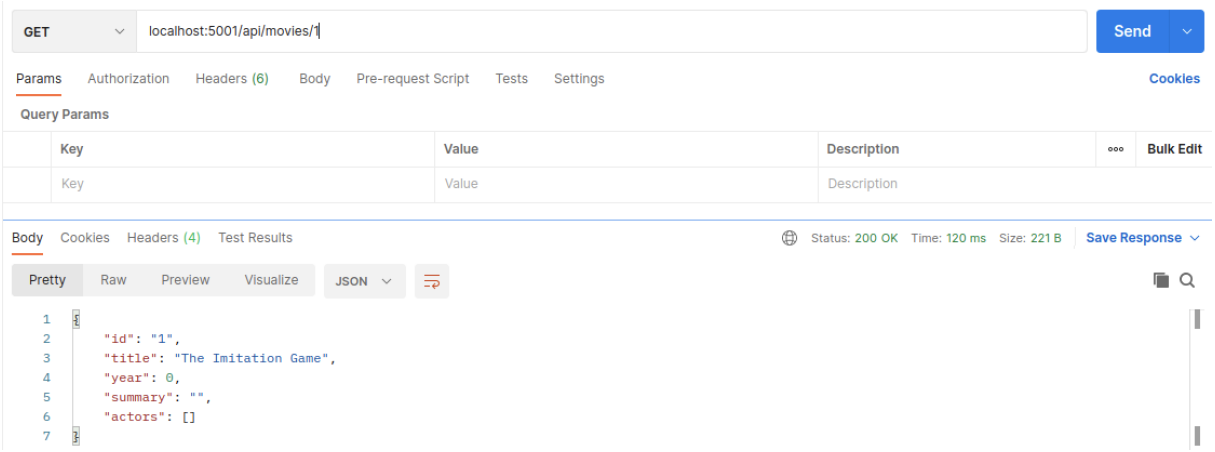



**Abbildung 1:** Postman GET-Request für Movie mit Id 1

> **Hinweis** Achten Sie darauf, dass die HTTP-Methode korrekt gesetzt ist. Wenn Sie im Request JSON-Daten übergeben, wählen sie in Tab Body die Option raw mit Typ JSON.

> **Tipp:** Wenn Sie einen Postman-Account haben, können Sie die verschiedenen Requests in Ihrem Postman-Workspace als Collection abspeichern.