MoviesApp\_RestAPI Project

API Document

Version 1.0

* Created By Manish Baddikuri

Table of Contents

[**ReadMe**: 2](#_Toc108877408)

[Swagger and Base URLs: 2](#_Toc108877409)

[Name 2](#_Toc108877410)

[URL 2](#_Toc108877411)

[BaseURL 2](#_Toc108877412)

[https://localhost:44316 2](#_Toc108877413)

[Swagger 2](#_Toc108877414)

[https://localhost:44316/swagger/index.html 2](#_Toc108877415)

[Introduction: 2](#_Toc108877416)

[Response structure: 3](#_Toc108877417)

[API’s Detail: 3](#_Toc108877418)

[Actor API’s: 3](#_Toc108877419)

[1. AddUpdateActor: 3](#_Toc108877420)

[2. GetActorsList 3](#_Toc108877421)

[3. GetActorsListDetail 4](#_Toc108877422)

[Producer API’s: 6](#_Toc108877423)

[4. AddUpdateProducer: 6](#_Toc108877424)

[5. GetProducersList 7](#_Toc108877425)

[6. GetProducersListDetail 8](#_Toc108877426)

[Movie API’s: 9](#_Toc108877427)

[7. AddUpdateMovie: 9](#_Toc108877428)

[8. AddUpdateMoviePoster: 9](#_Toc108877429)

[9. GetMoviePoster 9](#_Toc108877430)

[10. GetMoviesList 9](#_Toc108877431)

[11. GetMoviesListDetail 9](#_Toc108877432)

ReadMe:

Please perform the following tasks before running the Application

Step 1. Run the SQL script attached with the name “**SqlSqcript.sql**”

Step 2. Update the **connection sting** in the file “**AppSettings.json**” and replace the credentials (“Server” , “uid” , “pwd”).

All the Data is stored or created in the database with name “**MoviesApp\_RestAPI**”.

Swagger and Base URLs:

|  |  |
| --- | --- |
| Name | URL |
| BaseURL | https://localhost:44316 |
| Swagger | https://localhost:44316/swagger/index.html |

Introduction:

This document describes the Web API’s (Application Programming Interface) to programmatically access and exchange data with the MoviesApp\_RestAPI Project.

Web APIs will be provided as REST endpoints accessible over the internet using HTTPS protocol. All messages in the online API will be in JSON format.

* The following header must be provided in all API calls:

Content-Type: Header value should be “application/json” Path parameters and query parameters in the URL’s must be encoded using percentage encoding. (Refer https://www.w3schools.com/tags/ref\_urlencode.asp for details)

* Message Format: JSON

All request and response messages are in JSON (Javascript Object Notation) format. (Refer http://www.json.org/ for details).

* Some of the key specifications related to JSON and standards followed for the API’s are as follows.
  + JSON is built on 2 structures. Object or Map containing key value pairs and an ordered list of values.
  + A value could be Boolean (true / false), number, decimal, String or a structure (List or Object).
  + Object or key value pair structure consists of keys which are strings and values of any of the above types. E.g. {“name”:”Johan”, “age”:30}
  + List contains list of values. E.g. [“Johan”, “Mark”, “Lenin”]
  + A Boolean has only 2 values true or false.
  + String values are enclosed in single quote or double quotes. e.g. “name”, “Johan”, “Pending”
  + Numbers and decimals are represented without any thousand - separator character. Decimal indicator is dot (“.”)

Response structure:

All API’s will return the response with required data or message the form of a common response JSON structure.

Response JSON:

|  |  |  |  |
| --- | --- | --- | --- |
| **Parameter name** | **Data Type** | **Description** | **Meanings** |
| IsSuccess | boolean | Indicates Success or failure of request | True– Request Purpose Successful  False– Request Purpose failed |
| StatusCode | Int | Http Status Code indicating the status of request | 200/201 - Successful,  Other values - Failure |
| Message | String | Contains the message indicating success or the error message in case of request failure. | If IsSuccess is true, it will contain the success message.  If IsSuccess is false, it will contain the error message. |
| ResultData | JSON Object / String | Contains the result of the request.  Any data to be returned in the response shall be in this object. | Contains JSON result data if IsSuccess is true.  If IsSuccess is false, it will be null. |

API’s Detail:

Actor API’s:

Actor API’s deal with the Actor entity operations like adding, updating, deleting and also getting the list of actors.

### AddUpdateActor:

This API allows the user to add update or delete an Actor record

API Details:

|  |  |
| --- | --- |
| **API Name** | Actor/AddUpdateActor |
| **URL** | [BaseURL]/api/Actor/AddUpdateActor |
| **Method** | HTTP POST |
| **Content Type** | Application/json |
| **Response** | JSON |

Request Parameters:

|  |  |  |
| --- | --- | --- |
| Parameter name | Data type | Remarks |
| ActorId | int | Actor Id to update or delete, 0 in case of create record |
| ActorName | String |  |
| Bio | String |  |
| DOB | Date | Format: “yyyy-mm-dd” |
| Gender | String | Only fixed values are allowed:  “Male”,”Female”,”Others”. |
| ActivityType | Char | “C” – Create,  “U” – Update,  “D” - Delete |

Sample Request body:

|  |
| --- |
| {  "activityType": "C",  "actorName": "Johnson Down",  "bio": "Actor, Human",  "dob": "1970-01-25",  "gender": "Male"  }  [OR]  {  "activityType": "U",  "actorId": 4,  "actorName": "Scarlett Johansson",  "bio": "Actress, Hollywood",  "dob": "1985-10-25",  "gender": "feMale"  }  [OR]  {  "activityType": "D",  "actorId": 6  } |

Sample Response:

|  |
| --- |
| {  "isSuccess": true,  "statusCode": 201,  "message": "Actor Added successfully",  "resultData": null  }  [OR]  {  "isSuccess": false,  "statusCode": 400,  "message": "Actor Name already exists",  "resultData": null  } |

1. GetActorsList

This API allows the user to get a list of all the existing Actors (Ids and Actor Names)

API Details:

|  |  |
| --- | --- |
| **API Name** | Actor/GetActorsList |
| **URL** | [BaseURL]/api/Actor/GetActorsList |
| **Method** | HTTP GET |
| **Content Type** | Application/json |
| **Response** | JSON |

Request Parameters:

This API has no request parameters.

Sample Request:

|  |
| --- |
| [BaseURL]/api/Actor/GetActorsList |

Sample Response:

|  |
| --- |
| {  "isSuccess": true,  "statusCode": 200,  "message": "Success",  "resultData": [  {  "actorId": 1,  "actorName": "Robert Downey Jr."  },  {  "actorId": 2,  "actorName": "Scarlett Johansson"  },  {  "actorId": 3,  "actorName": "Luke Skywalker"  }  ]  } |

1. GetActorsListDetail

This API allows the user to get a list of all the existing Actors.

API Details:

|  |  |
| --- | --- |
| **API Name** | Actor/GetActorsList |
| **URL** | [BaseURL]/api/Actor/GetActorsListDetail |
| **Method** | HTTP GET |
| **Content Type** | Application/json |
| **Response** | JSON |

Request Parameters:

This API has no request parameters.

Sample Request:

|  |
| --- |
| [BaseURL]/api/Actor/ GetActorsListDetail |

Sample Response:

|  |
| --- |
| {  "isSuccess": true,  "statusCode": 200,  "message": "Success",  "resultData": [  {  "actorId": 1,  "actorName": "Robert Downey Jr.",  "bio": "Iron Man AKA Tony Stark. One of the most famous actors.",  "dob": "1965-04-04T00:00:00",  "gender": "Male"  },  {  "actorId": 2,  "actorName": "Scarlett Johansson",  "bio": "One of the most famous actress.",  "dob": "1975-05-20T00:00:00",  "gender": "Female"  },  {  "actorId": 3,  "actorName": "Luke Skywalker",  "bio": "A new hope, Son of Anakin Skywalker.",  "dob": "1977-10-05T00:00:00",  "gender": "Male"  }  ]  } |

Producer API’s:

Producer API’s deal with the Producer entity operations like adding, updating, deleting and also getting the list of actors.

1. AddUpdateProducer:

This API allows the user to create a new Producer record

API Details:

|  |  |
| --- | --- |
| **API Name** | Producer/AddUpdateProducer |
| **URL** | [BaseURL]/api/Producer/AddUpdateProducer |
| **Method** | HTTP POST |
| **Content Type** | Application/json |
| **Response** | JSON |

Request Parameters:

|  |  |  |
| --- | --- | --- |
| Parameter name | Data type | Remarks |
| ProducerId | int | Producer Id to update or delete, 0 in case of create record |
| ProducerName | String |  |
| Bio | String |  |
| DOB | Date | Format: “yyyy-mm-dd” |
| Gender | String | Only fixed values are allowed:  “Male”,”Female”,”Others”. |
| Company | String |  |
| ActivityType | Char | “C” – Create,  “U” – Update,  “D” - Delete |

Sample Request body:

|  |
| --- |
| {  "activityType": "C",  "producerName": "Kevin Feige",  "bio": "The man behind world's most successful movie franchise",  "dob": "1975-03-27",  "gender": "Male",  "company": "Marvel Cinematic Universe"  }  [OR]  {  "activityType": "U",  "producerId": 1,  "producerName": "Kevin Feige",  "bio": "The man behind one of the world's most successful movie franchise",  "dob": "1975-03-27",  "gender": "Male",  "company": "Marvel Cinematic Universe"  }  [OR]  {  "activityType": "D",  "producerId": 4  } |

Sample Response:

|  |
| --- |
| {  "isSuccess": true,  "statusCode": 201,  "message": "Producer Added successfully",  "resultData": null  }  [OR]  {  "isSuccess": false,  "statusCode": 400,  "message": "Producer Name already exists",  "resultData": null  } |

1. GetProducersList

This API allows the user to get a list of all the existing Producers (Ids and Producer Names)

API Details:

|  |  |
| --- | --- |
| **API Name** | Producer/GetProducersList |
| **URL** | [BaseURL]/api/Producer/GetProducersList |
| **Method** | HTTP GET |
| **Content Type** | Application/json |
| **Response** | JSON |

Request Parameters:

This API has no request parameters.

Sample Request:

|  |
| --- |
| [BaseURL]/api/Producer/GetProducersList |

Sample Response:

|  |
| --- |
| {  "isSuccess": true,  "statusCode": 200,  "message": "Success",  "resultData": [  {  "producerId": 1,  "producerName": "Kevin Feige"  },  {  "producerId": 2,  "producerName": "Jon Favreau"  },  {  "producerId": 3,  "producerName": "George Lucas"  }  ]  } |

1. GetProducersListDetail

This API allows the user to get a list of all the existing Producers.

API Details:

|  |  |
| --- | --- |
| **API Name** | Producer/GetProducersList |
| **URL** | [BaseURL]/api/Producer/GetProducersListDetail |
| **Method** | HTTP GET |
| **Content Type** | Application/json |
| **Response** | JSON |

Request Parameters:

This API has no request parameters.

Sample Request:

|  |
| --- |
| [BaseURL]/api/Producer/ GetProducersListDetail |

Sample Response:

|  |
| --- |
| {  "isSuccess": true,  "statusCode": 200,  "message": "Success",  "resultData": [  {  "producerId": 1,  "producerName": "Kevin Feige",  "bio": "The man behind world's most successful movie franchise",  "dob": "1972-03-27T00:00:00",  "gender": "Male",  "company": "Marvel Cinematic Universe"  },  {  "producerId": 2,  "producerName": "Jon Favreau",  "bio": "The one who is behind almost every movie franchise with new concepts.",  "dob": "1970-03-27T00:00:00",  "gender": "Male",  "company": "Disney"  },  {  "producerId": 3,  "producerName": "George Lucas",  "bio": "Creator of StarWars",  "dob": "1965-03-27T00:00:00",  "gender": "Male",  "company": "StarWars"  }  ]  } |

Movie API’s:

Movie API’s deal with the Movie entity operations like adding, updating, deleting and also getting the list of actors.

1. AddUpdateMovie:

This API allows the user to add update or delete a Movie record

API Details:

|  |  |
| --- | --- |
| **API Name** | Movie/AddUpdateMovie |
| **URL** | [BaseURL]/api/Movie/AddUpdateMovie |
| **Method** | HTTP POST |
| **Content Type** | Application/json |
| **Response** | JSON |

Request Parameters:

|  |  |  |
| --- | --- | --- |
| Parameter name | Data type | Remarks |
| MovieId | Int | Id of the movie to update or delete, 0 in case of create record |
| MovieName | String | Mandatory to create a movie |
| ProducerId | Int | ProducerId |
| ActorIds | Array of Integers | Must contain the ActorIds of Actors associated with the movie |
| Plot | String |  |
| DateOfRelease | Date | Format: “yyyy-mm-dd” |
| ActivityType | Char | “C” – Create,  “U” – Update,  “D” - Delete |

Sample Request body:

|  |
| --- |
| {  "activityType": "C",  "movieName": "Rise of the Planet of Robots",  "producerId": 2,  "actorIds": [3,2],  "plot": "Robots learn and decide to rise",  "dateOfRelease": "2022-05-08"  }  [OR]  {  "movieId": 2,  "activityType": "U",  "movieName": "Rise of the Planet of Robots - III",  "producerId": 2,  "actorIds": [4,2],  "plot": "Robots rise to power after a decision",  "dateOfRelease": "2022-05-08"  }  [OR]  {  "movieId": 3,  "activityType": "D"  } |

Sample Response:

|  |
| --- |
| {  "isSuccess": true,  "statusCode": 201,  "message": "Movie Added successfully",  "resultData": null  }  [OR]  {  "isSuccess": false,  "statusCode": 400,  "message": "Movie Name already exists",  "resultData": null  } |

1. AddUpdateMoviePoster:

This API allows the user to add update or delete Movie Poster

API Details:

|  |  |
| --- | --- |
| **API Name** | Movie/AddUpdateMoviePoster |
| **URL** | [BaseURL]/api/Movie/AddUpdateMoviePoster |
| **Method** | HTTP POST |
| **Content Type** | Application/json |
| **Response** | JSON |

Request Parameters:

|  |  |  |
| --- | --- | --- |
| Parameter name | Data type | Remarks |
| MovieId | Int | Id of the movie to Add, update or delete Poster, 0 in case of create record |
| moviePoster | String | String encoded in base 64 converted from bytes of image |
| ActivityType | Char | “C” – Create,  “U” – Update,  “D” - Delete |

Sample Request body:

|  |
| --- |
| {  "activityType": "C",  "movieId": 2,  "moviePoster": "/9j/4AAQSkZJRgABAQAAAQABAAD/2wCEAAkGBxMTEhUTExMWFRUXGBcYGBgYFxgXHRcYFxcaFxsXGBcaHSgiGBolHxgXITEhJSkrLi4uGB8zODMsNygtLisBCgoKDg0OGxAQGysmICYvLS8tLS0vLS0tNSstLS0vLTUtLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS01LS0tLf/AABEIAREAuAMBIgACEQEDEQH/xAAcAAABBQEBAQAAAAAAAAAAAAAFAAIDBAYBBwj/ "  }  [OR]  {  "activityType": "U",  "movieId": 3,  "moviePoster": "/9j/4AAQSkZJRgABAQAAAQABAAD/2wCEAAkGBxMTEhUTExMWFRUXGBcYGBgYFxgXHRcYFxcaFxsXGBcaHSgiGBolHxgXITEhJSkrLi4uGB8zODMsNygtLisBCgoKDg0OGxAQGysmICYvLS8tLS0vLS0tNSstLS0vLTUtLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS01LS0tLf/AABEIAREAuAMBIgACEQEDEQH/xAAcAAABBQEBAQAAAAAAAAAAAAAFAAIDBAYBBwj/ "  } [OR]  {  "activityType": "D",  "movieId": 2  } |

Sample Response:

|  |
| --- |
| {  "isSuccess": true,  "statusCode": 201,  "message": "Movie Poster Added successfully",  "resultData": null  }  [OR]  {  "isSuccess": false,  "statusCode": 400,  "message": "Movie Poster Already exists for this MovieId.",  "resultData": null  }  [OR]  {  "isSuccess": true,  "statusCode": 201,  "message": "Movie Poster Deleted successfully",  "resultData": null  } |

1. GetMoviePoster

This API allows the user to get a list of all the existing Movies (Ids and Movie Names)

API Details:

|  |  |
| --- | --- |
| **API Name** | Movie/GetMoviePoster |
| **URL** | [BaseURL]/api/Movie/GetMoviePoster{MovieId} |
| **Method** | HTTP GET |
| **Content Type** | Application/json |
| **Response** | JSON |

Request Parameters:

MovieId must be supplied in the Query String of the API

Sample Request:

|  |
| --- |
| [BaseURL]/api/Movie/GetMoviePoster/{movieId}  [OR]  [BaseURL]/api/Movie/GetMoviePoster/2 |

Sample Response:

|  |
| --- |
| {  "isSuccess": true,  "statusCode": 201,  "message": "Success",  "resultData": [  {  "movieId": 2,  "poster": "/9j/4AAQSkZJRgABAQAAAQABAAD/2wCEAAkGBxMTEhUTExMWFRUXGBcYGBgYFxgXHRcYFxcaFxsXGBcaHSgiGBolHxgXITEhJSkrLi4uGB8zODMsNygtLisBCgoKDg0OGxAQGysmICYvLS8tLS0vLS0tNSstLS0vLTUtLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS0tLS01LS0tLf/AABEIAREAuAMBIgACEQEDEQH/xAAcAAABBQEBAQAAAAAAAAAAAAAFAAIDBAYBBwj/ "  }  ]  } |

1. GetMoviesList

This API allows the user to get a list of all the existing Movies (Ids and Movie Names)

API Details:

|  |  |
| --- | --- |
| **API Name** | Movie/GetMoviesList |
| **URL** | [BaseURL]/api/Movie/GetMoviesList |
| **Method** | HTTP GET |
| **Content Type** | Application/json |
| **Response** | JSON |

Request Parameters:

This API has no request parameters.

Sample Request:

|  |
| --- |
| [BaseURL]/api/Movie/GetMoviesList |

Sample Response:

|  |
| --- |
| {  "isSuccess": true,  "statusCode": 200,  "message": "Success",  "resultData": [  {  "movieId": 1,  "movieName": "Rise of the Planet of Robots"  },  {  "movieId": 3,  "movieName": "Cyclotron Returns"  },  {  "movieId": 4,  "movieName": "CapsLock Shift"  },  {  "movieId": 7,  "movieName": "Shift Nitro"  }  ]  } |

1. GetMoviesListDetail

This API allows the user to get a list of all the existing Movies.

API Details:

|  |  |
| --- | --- |
| **API Name** | Movie/ GetMoviesListDetail |
| **URL** | [BaseURL]/api/Movie/GetMoviesListDetail |
| **Method** | HTTP GET |
| **Content Type** | Application/json |
| **Response** | JSON |

Request Parameters:

This API has no request parameters.

Sample Request:

|  |
| --- |
| [BaseURL]/api/Movie/ GetMoviesListDetail |

Sample Response:

|  |
| --- |
| {  "isSuccess": true,  "statusCode": 200,  "message": "Success",  "resultData": [  {  "movieId": 1,  "movieName": "Rise of the Planet of Robots",  "actorIds": null,  "plot": "Robots learn and decide to rise",  "dateOfRelease": "2022-05-08T00:00:00",  "actorsList": [  {  "actorId": 1,  "actorName": "Robert Downey Jr."  },  {  "actorId": 2,  "actorName": "Scarlett Johansson"  }  ],  "producerId": 2,  "producerName": "Jon Favreau",  },  {  "movieId": 3,  "movieName": "Cyclotron Returns",  "actorIds": null,  "plot": "Cyclotron returns to claim its planet",  "dateOfRelease": "2022-01-08T00:00:00",  "actorsList": [  {  "actorId": 1,  "actorName": "Robert Downey Jr."  },  {  "actorId": 2,  "actorName": "Scarlett Johansson"  }  ],  "producerId": 1,  "producerName": "Kevin Feige",  },  {  "movieId": 4,  "movieName": "CapsLock Shift",  "actorIds": null,  "plot": "With time running the guy must find capslock",  "dateOfRelease": "2021-10-21T00:00:00",  "actorsList": [  {  "actorId": 2,  "actorName": "Scarlett Johansson"  }  ],  "producerId": 3,  "producerName": "George Lucas",  },  {  "movieId": 7,  "movieName": "Shift Nitro",  "actorIds": null,  "plot": "With time running the guy must find capslock",  "dateOfRelease": "2021-10-21T00:00:00",  "actorsList": [  {  "actorId": 5,  "actorName": "The Guy"  },  {  "actorId": 4,  "actorName": "Luke Skywalker"  }  ],  "producerId": 2,  "producerName": "Jon Favreau",  }  ]  } |