

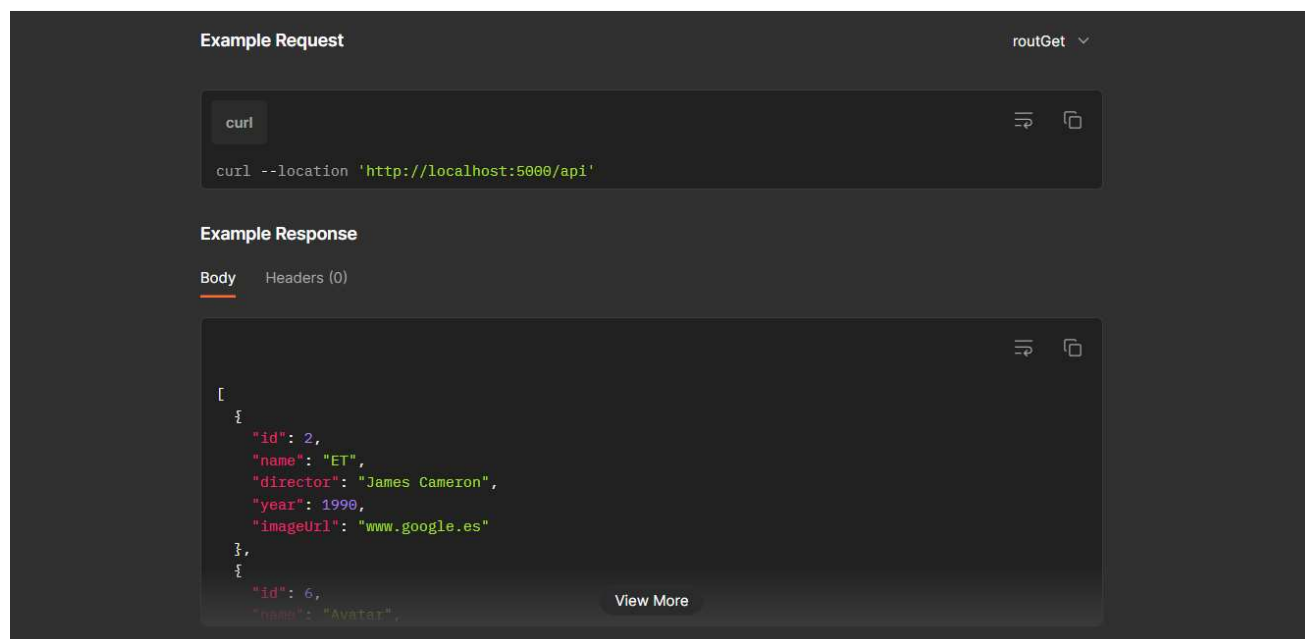
routes-HollywowApi

Methods in Postman to view, add, edit, and delete movie posters in our database.

GET routGet

```
http://localhost:5000/api
```

Route to do the method GET from the CRUD of our project, including our URL and an example of the response.

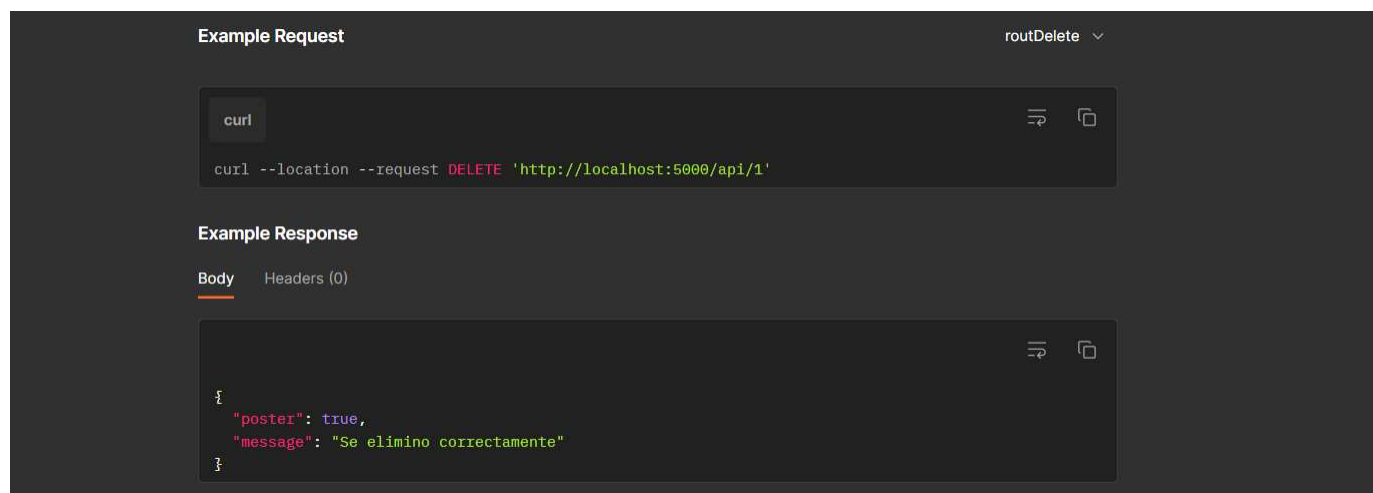


The screenshot shows the Postman interface for the GET method. The 'Example Request' section displays a curl command: `curl --location 'http://localhost:5000/api'`. The 'Example Response' section shows a JSON array of movie objects. The first object is: `{ "id": 2, "name": "ET", "director": "James Cameron", "year": 1990, "imageUrl": "www.google.es" }`. The second object is partially visible: `{ "id": 6, "name": "Avatar",`. A 'View More' link is present at the bottom of the response area.

DELETE routDelete

```
http://localhost:5000/api/1
```

Route to do the method DELETE from the CRUD of our project, including our URL and an example of the response.



The screenshot shows the Postman interface for the DELETE method. The 'Example Request' section displays a curl command: `curl --location --request DELETE 'http://localhost:5000/api/1'`. The 'Example Response' section shows a JSON object: `{ "poster": true, "message": "Se elimino correctamente" }`.

PUT routPut

http://localhost:5000/api/1

Route to do the method PUT from the CRUD of our project, including our URL and an example of the response.

Body raw (json)

```
json

{
  "name": "",
  "director": "",
  "year": 1990,
  "imageUrl": ""
}
```

Example Request

curl

```
curl --location --request PUT 'http://localhost:5000/api/6' \
--data '{
  "name": "Avatar",
  "director": "James Cameron",
  "year": 1990,
  "imageUrl": "www.google.es"
}'
```

Example Response

Body Headers (0)

```
{
  "poster": {
    "id": 6,
    "name": "Avatar",
    "director": "James Cameron",
    "year": 1990,
    "imageUrl": "www.google.es"
  },
  "message": "Se actualizó el poster correctamente"
}
```

POST routPost

http://localhost:5000/api

Route to do the method POST from the CRUD of our project, including our URL and an example of the response.

Body raw (json)

```
json

{
  "name": "",
  "director": "",
  "year": 1990,
  "imageUrl": ""
}
```

The screenshot displays a REST client interface with two main sections: "Example Request" and "Example Response".

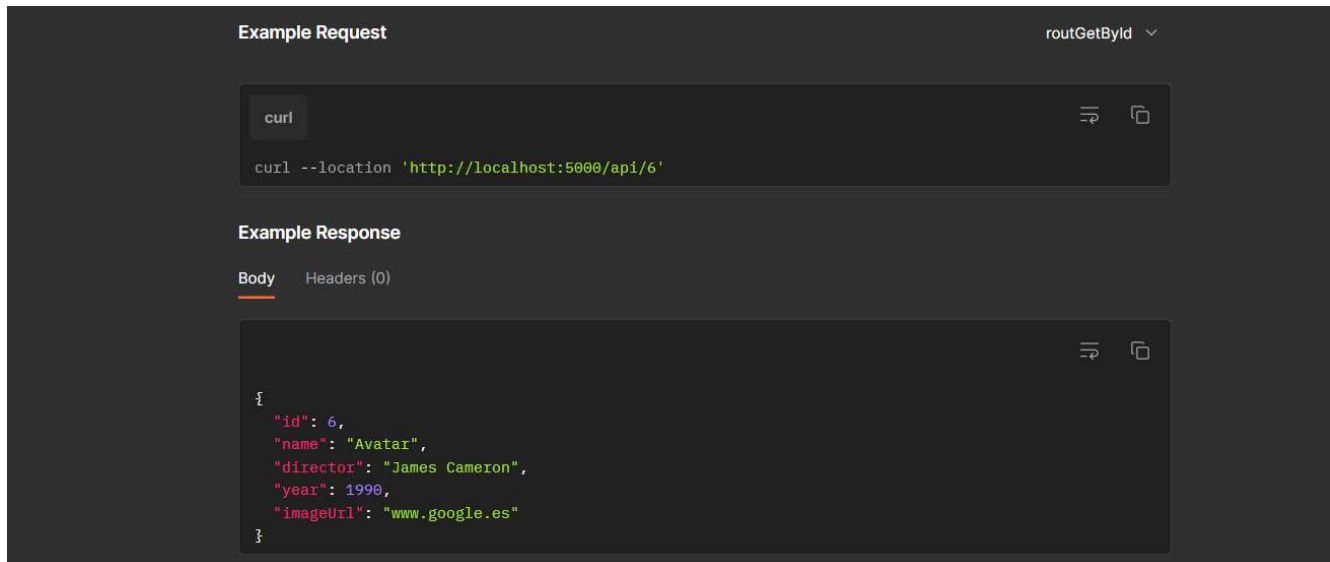
Example Request: The request is a curl command targeting the URL `http://localhost:5000/api`. The data body is a JSON object: `{ "name": "Eduardo Manostijeras", "director": "Tim Burton", "year": 1990, "imageUrl": "https://encrypted-tbn2.gstatic.com/images?q=tbn:AND9GcQ3GL6hYh3mNwP7vhaSmj5VUU3:" }`.

Example Response: The response is a JSON object with the following structure: `{ "poster": { "id": 42, "name": "Eduardo Manostijeras", "director": "Tim Burton", "year": 1990, "imageUrl": "https://encrypted-tbn2.gstatic.com/images?q=tbn:AND9GcQ3GL6hYh3mNwP7vhaSmj5VUU3:irYf"}, "message": "El poster se creo con éxito" }`.

GET routGetById

http://localhost:5000/api/1

Route to do the method GET BY ID from the CRUD of our project, including our URL and an example of the response.



The screenshot displays a REST client interface with a dark theme. At the top right, the endpoint `routGetById` is selected. The **Example Request** section shows a `curl` command: `curl --location 'http://localhost:5000/api/6'`. Below this, the **Example Response** section is active, showing the response body as a JSON object: `{ "id": 6, "name": "Avatar", "director": "James Cameron", "year": 1990, "imageUrl": "www.google.es" }`. The response body is highlighted with a light blue background.