

## Add movie

The screenshot shows the Postman interface with a workspace named "My Workspace". The left sidebar displays a "Collections" view with several collections, including "ujianbackend" which is currently selected. The main panel shows a POST request to the endpoint `{{url}}/addmovies`. The request body is a JSON object:

```
1 {
2   "nama": "Avengers: End Game",
3   "tahun": 2019,
4   "deskript": "Sedih banget dijamin mevveque"
5 }
```

The response is displayed in the "Body" tab, showing a JSON object with the following fields:

```
1 {
2   "fieldCount": 0,
3   "affectedRows": 1,
4   "insertId": 2,
5   "serverStatus": 2,
6   "warningCount": 0,
7   "message": "",
8   "protocol41": true,
9   "changedRows": 0
10 }
```

The status bar at the bottom indicates a successful response with status 200 OK, time 45 ms, and size 340 B.

## Edit movie

The screenshot shows the Postman interface with a workspace named "My Workspace". The left sidebar displays a "Collections" view with several collections, including "ujianbackend" which is currently selected. The main panel shows a PATCH request to the endpoint `{{url}}/editmovies/:id`. The request parameters are:

KEY	VALUE	DESCRIPTION
id	5	Description

The response is displayed in the "Response" tab, showing a message: "Hit the Send button to get a response."

My Workspace

ujianbackend

Filter

History Collections

Trash

finalproject 8 requests

cinema\_booking 5 requests

jc8ExpressMysql 4 requests

jc8ReactMongoose 9 requests

ujianbackend 10 requests

Categories

POST Add category

PATCH Edit category

DEL Delete category

GET Show all categories

Movies

POST Add movies

PATCH Edit movies

DEL Delete movies

GET Show all movies

CatMov

PATCH Edit movies

DELETES

Send Save

Params Authorization Headers (9) Body Pre-request Script Tests

none form-data x-www-form-urlencoded raw binary JSON (application/json)

1 2 3 4 5

```
1 {
2   "nama": "Game of Thrones S8",
3   "tahun": 2019,
4   "deskripst": "Seru bat bikin hati terbabat"
5 }
```

Body Cookies Headers (6) Test Results

Status: 200 OK Time: 15 ms Size: 381 B Save Download

Pretty Raw Preview JSON

```
1 {
2   "fieldCount": 0,
3   "affectedRows": 1,
4   "insertId": 0,
5   "serverStatus": 2,
6   "warningCount": 0,
7   "message": "(Rows matched: 1 Changed: 0 Warnings: 0",
8   "protocol41": true,
9   "changedRows": 0
10 }
```

## Delete movie

My Workspace

ujianbackend

Filter

History Collections

Trash

finalproject 8 requests

cinema\_booking 5 requests

jc8ExpressMysql 4 requests

jc8ReactMongoose 9 requests

ujianbackend 3 requests

POST Add movies

GET Edit movies

GET Delete movies

POST Add movies

PATCH Edit movies

DEL Delete movies

DELETES

Send Save

Params Authorization Headers (8) Body Pre-request Script Tests

Query Params

KEY	VALUE	DESCRIPTION	Bulk Edit
Key	Value	Description	

Path Variables

KEY	VALUE	DESCRIPTION	Bulk Edit
id	4	Description	

Body Cookies Headers (6) Test Results

Status: 200 OK Time: 15 ms Size: 340 B Save Download

Pretty Raw Preview JSON

```
1 {
2   "fieldCount": 0,
3   "affectedRows": 1,
4   "insertId": 0,
5   "serverStatus": 2,
6   "warningCount": 0,
7   "message": "",
8   "protocol41": true,
9   "changedRows": 0
10 }
```

## Show all movies

The screenshot shows the Postman interface with a GET request to 'Show all movies' selected. The request is configured with the method 'GET' and the URL '({url})/allmovies'. The response is displayed in the 'Body' tab, showing a JSON array of two movie objects. The status is 200 OK, with a time of 71 ms and a size of 385 B.

**Request:**

- Method: GET
- URL: ({url})/allmovies

**Response Body:**

```
1 - {
2 -   {
3 -     "id": 1,
4 -     "nama": "Avengers: End Game",
5 -     "tahun": 2019,
6 -     "deskripsi": "Sedih bat dijamin mevveque"
7 -   },
8 -   {
9 -     "id": 2,
10 -    "nama": "Avengers: Infinity War",
11 -    "tahun": 2018,
12 -    "deskripsi": "Thegang coy"
13 -   }
14 - }
```

## Add categories

The screenshot shows the Postman interface with a POST request to 'Add category' selected. The request is configured with the method 'POST' and the URL '({url})/addcategories'. The request body is set to 'raw' with the JSON content '{"nama": "Drama"}'. The response is displayed in the 'Body' tab, showing a JSON object with details about the inserted category. The status is 200 OK, with a time of 85 ms and a size of 340 B.

**Request:**

- Method: POST
- URL: ({url})/addcategories
- Body: raw (JSON application/json)
- Content: {"nama": "Drama"}

**Response Body:**

```
1 - {
2 -   "fieldCount": 0,
3 -   "affectedRows": 1,
4 -   "insertId": 2,
5 -   "serverStatus": 2,
6 -   "warningCount": 0,
7 -   "message": "",
8 -   "protocol41": true,
9 -   "changedRows": 0
10 - }
```

## Edit categories

The screenshot shows the Postman interface with the 'Edit category' PATCH request configured. The left sidebar displays a collection named 'ujianbackend' with a sub-collection 'Categories'. The main panel shows the request details for the 'Edit category' endpoint.

**Request Configuration:**

- Method:** PATCH
- URL:** `{{url}}/editcategories/id`
- Params:** One query parameter is defined: 

KEY	VALUE	DESCRIPTION
Key	Value	Description
- Path Variables:** One path variable is defined: 

KEY	VALUE	DESCRIPTION
id	2	Description
- Response:** A message indicates: "Hit the Send button to get a response."

The screenshot shows the Postman interface with the 'Edit category' PATCH request executed. The left sidebar displays the 'ujianbackend' collection. The main panel shows the response details for the 'Edit category' endpoint.

**Request Configuration (Top):**

- Method:** PATCH
- URL:** `{{url}}/editcategories/id`
- Params:** One query parameter is defined: 

KEY	VALUE	DESCRIPTION
Key	Value	Description
- Path Variables:** One path variable is defined: 

KEY	VALUE	DESCRIPTION
id	2	Description
- Response:** A message indicates: "Hit the Send button to get a response."

**Response Details (Bottom):**

- Status:** 200 OK
- Time:** 13 ms
- Size:** 381 B
- Body:** The response body is displayed in JSON format: 

```
1 {
2   "fieldCount": 0,
3   "affectedRows": 1,
4   "insertId": 0,
5   "serverStatus": 2,
6   "warningCount": 0,
7   "message": "(Rows matched: 1 Changed: 1 Warnings: 0",
8   "protocol41": true,
9   "changedRows": 1
10 }
```

## Delete categories

The screenshot shows the Postman interface with a workspace named "My Workspace". The left sidebar displays a collection of requests under the "ujianbackend" folder, including "Categories" with sub-items "Add category", "Edit category", and "Delete category". The "Delete category" request is selected.

The main panel shows the details of the "Delete category" request:

- Method:** DELETE
- URL:** `{{url}}/deletecategories/id`
- Params:** A table with one entry: 

KEY	VALUE	DESCRIPTION
id	2	Description
- Body:** The response is displayed in JSON format: 

```
1 {
2   "fieldCount": 0,
3   "affectedRows": 1,
4   "insertId": 0,
5   "serverStatus": 2,
6   "warningCount": 0,
7   "message": "",
8   "protocol41": true,
9   "changedRows": 0
10 }
```
- Status:** 200 OK, Time: 65 ms, Size: 340 B

## Show all category

The screenshot shows the Postman interface with a workspace named "My Workspace". The left sidebar displays a collection of requests under the "ujianbackend" folder, including "Categories" with sub-items "Add category", "Edit category", "Delete category", and "Show all categories". The "Show all categories" request is selected.

The main panel shows the details of the "Show all categories" request:

- Method:** GET
- URL:** `{{url}}/allcategories`
- Params:** A table with one entry: 

KEY	VALUE	DESCRIPTION
Key	Value	Description
- Body:** The response is displayed in JSON format: 

```
1 [
2   {
3     "id": 1,
4     "nama": "Action"
5   }
6 ]
```
- Status:** 200 OK, Time: 52 ms, Size: 238 B

## Add connection

The screenshot shows the Postman interface with a workspace named 'My Workspace'. On the left sidebar, the 'Collections' tab is active, showing a collection named 'ujianbackend' with 9 requests. The 'CatMov' sub-collection is expanded, showing a 'POST Add conn' request. The main panel displays the details of this request:

- Method:** POST
- URL:** {{url}}/addconnection
- Body:** The 'Body' tab is selected, showing a JSON payload:

```
1- {
2-   "movie_id": 2,
3-   "category_id": 1
4- }
```
- Status:** 200 OK, Time: 117 ms, Size: 338 B
- Response:** The 'Body' tab shows the response in JSON format:

```
1- [
2-   {
3-     "nama_film": "Avengers: End Game",
4-     "nama_kategori": "Action"
5-   },
6-   {
7-     "nama_film": "Avengers: Infinity War",
8-     "nama_kategori": "Action"
9-   }
10- ]
```

## Delete connection

The screenshot shows the Postman interface with the same workspace. The 'Delete connection' request is selected in the 'CatMov' sub-collection. The main panel displays the details of this request:

- Method:** DELETE
- URL:** {{url}}/deleteconnection/id
- Params:** The 'Params' tab is selected, showing query parameters:

KEY	VALUE	DESCRIPTION
id	2	Description
- Status:** 200 OK, Time: 109 ms, Size: 273 B
- Response:** The 'Body' tab shows the response in JSON format:

```
1- [
2-   {
3-     "nama_film": "Avengers: End Game",
4-     "nama_kategori": "Action"
5-   }
6- ]
```