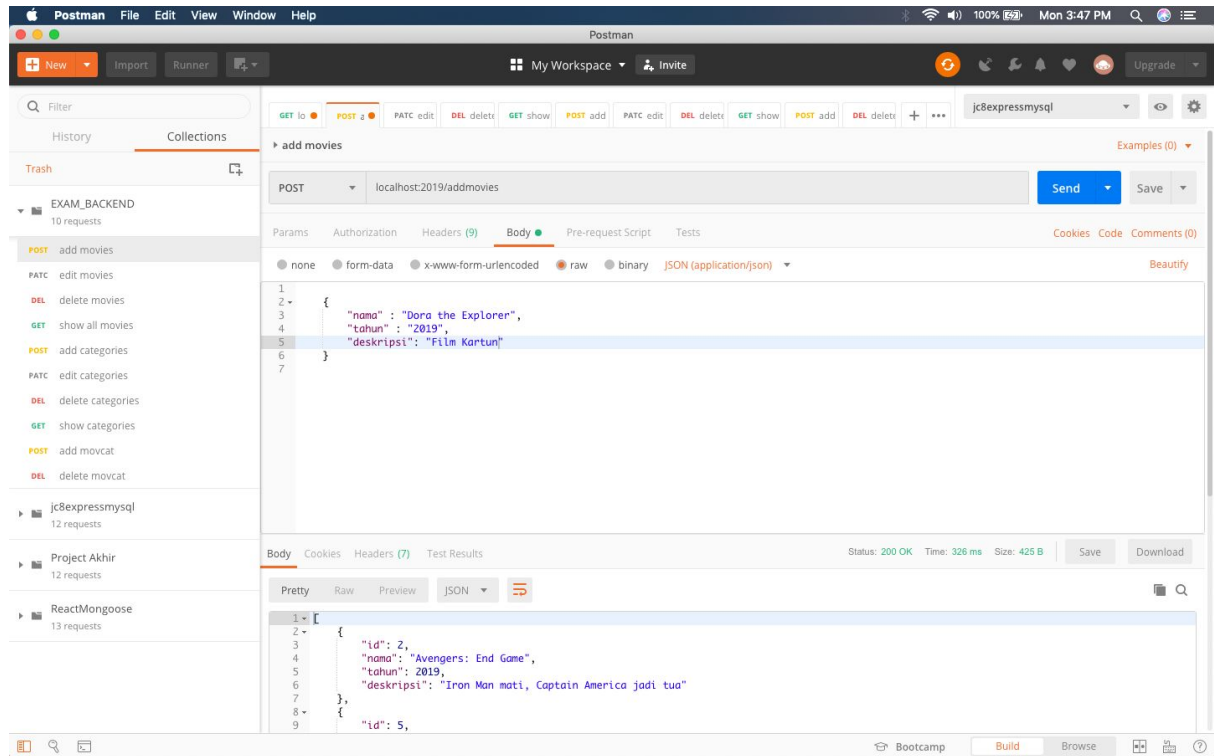


Dokumentasi POSTMAN

ADD MOVIES



EDIT MOVIES

req.params

The screenshot shows the Postman application with a PATCH request to `localhost:2019/editmovies/moviesid`. The request is configured with the following parameters:

- Query Params:**
 - Key: `key`, Value: `Value`, Description: `Description`
- Path Variables:**
 - Key: `moviesid`, Value: `1`, Description: `Description`

The response body is displayed in the 'Pretty' view, showing a JSON array of movie objects:

```
1- [
2-   {
3-     "id": 1,
4-     "nama": "Transformer",
5-     "tahun": 2010,
6-     "deskripsi": "bolehlah"
7-   },
8-   {
9-     "id": 2,
10-    "nama": "Avengers: End Game",
11-    "tahun": 2019,
12-    "deskripsi": "Iron Man mati, Captain America jadi tua"
13-  }
14- ]
```

req.body

The screenshot shows the Postman application with a PATCH request to `localhost:2019/editmovies/moviesid`. The request is configured with the following parameters:

- Body:** The body is set to 'raw' and contains the following JSON:

```
1- {
2-   "nama": "Transformer",
3-   "tahun": "2010",
4-   "deskripsi": "bolehlah"
5- }
```

The response body is displayed in the 'Pretty' view, showing a JSON array of movie objects:

```
1- [
2-   {
3-     "id": 1,
4-     "nama": "Transformer",
5-     "tahun": 2010,
6-     "deskripsi": "bolehlah"
7-   },
8-   {
9-     "id": 2,
```

DELETE MOVIES

The screenshot shows the Postman application interface. On the left, a sidebar lists collections: EXAM_BACKEND (10 requests), jc8expressmysql (12 requests), Project Akhir (12 requests), and ReactMongoose (13 requests). The 'delete movies' request is selected under EXAM_BACKEND. The main panel shows the request details for 'delete movies' with a DELETE method and URL 'localhost:2019/deletemovies/moviesid'. The 'Params' tab is active, showing a query parameter 'moviesid' with value '1'. The 'Body' tab is also active, showing a JSON response body. The status is 200 OK, time is 546 ms, and size is 350 B. The response body is displayed in the 'Pretty' view.

DELETE localhost:2019/deletemovies/moviesid

Params

KEY	VALUE	DESCRIPTION
moviesid	1	Description

Body

```
1 [
2   {
3     "id": 2,
4     "nama": "Avengers: End Game",
5     "tahun": 2019,
6     "deskripsi": "Iron Man mati, Captain America jadi tua"
7   }
8 ]
```

SHOW ALL MOVIES

The screenshot shows the Postman application interface. The left sidebar displays a collection named 'EXAM_BACKEND' with 10 requests. The 'show all movies' request is selected, which is a GET request to 'localhost:2019/showmovies'. The main panel shows the request details, including the URL, method (GET), and the response body in JSON format. The response body contains an array of two movie objects.

Request Details:

- Method: GET
- URL: localhost:2019/showmovies
- Status: 200 OK
- Time: 64 ms
- Size: 425 B

Response Body (JSON):

```
[{"id": 2, "nama": "Avengers: End Game", "tahun": 2019, "deskripsi": "Iron Man mati, Captain America jadi tua"}, {"id": 5, "nama": "Dora the Explorer", "tahun": 2019, "deskripsi": "Film Kartun"}]
```

ADD CATEGORIES

The screenshot shows the Postman application interface. The top bar includes the Postman logo, menu items (File, Edit, View, Window, Help), and system status (100% zoom, Mon 4:02 PM). The main workspace is titled 'My Workspace' and 'Invite'. The left sidebar shows a 'Collections' view with a search filter and a list of collections: EXAM_BACKEND (10 requests), jc8expressmysql (12 requests), Project Akhir (12 requests), and ReactMongoose (13 requests). The 'add categories' collection is expanded, showing a list of requests. The 'POST' request 'add categories' is selected. The main panel shows the details of this request. The URL is 'localhost:2019/categories'. The method is 'POST'. The body is a JSON array with one object: { 'id': 1, 'name': 'Action' }. The response is a JSON array with two objects: { 'id': 1, 'name': 'Korean Film' } and { 'id': 2, 'name': 'Drama' }. The status is '200 OK', time is '164 ms', and size is '350 B'. The response is displayed in the 'Body' tab, which is currently set to 'Pretty' view.

POST localhost:2019/categories

Params Authorization Headers (9) Body Pre-request Script Tests Cookies Code Comments (0)

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

```
1 {
2   "name": "Action"
3 }
```

Status: 200 OK Time: 164 ms Size: 350 B Save Download

Pretty Raw Preview JSON

```
1 [
2   {
3     "id": 1,
4     "name": "Korean Film"
5   },
6   {
7     "id": 2,
8     "name": "Drama"
9   }
]
```

EDIT CATEGORIES

req. params

The screenshot shows the Postman application with a PATCH request configured. The URL is `localhost:2019/editcategories/categoriesid`. The 'Params' tab is active, showing a single query parameter: `categoriesid` with a value of `1`. The 'Body' tab is also visible, showing a JSON array of category objects. The status bar at the bottom indicates a 200 OK response with a time of 257 ms and a size of 345 B.

KEY	VALUE	DESCRIPTION
categoriesid	1	Description

```
1- [
2-   {
3-     "id": 1,
4-     "nama": "Sci-fi"
5-   },
6-   {
7-     "id": 2,
8-     "nama": "Drama"
9-   },
10-  {
11-    "id": 3,
12-    "nama": "Comedy"
13-  },
14-  {
15-    "id": 5,
16-    "nama": "Action"
17-  }
18- ]
```

req. body

The screenshot shows the Postman application with the same PATCH request. The 'Body' tab is active, and the 'raw' radio button is selected. The body content is a JSON object: `{ "nama": "Sci-fi" }`. The 'JSON (application/json)' format is selected. The status bar at the bottom indicates a 200 OK response with a time of 257 ms and a size of 345 B.

```
1- {
2-   "nama": "Sci-fi"
3- }
```

DELETE CATEGORIES

Postman

File Edit View Window Help

Postman

New Import Runner

My Workspace Invite

100% Mon 4:03 PM

Filter

History Collections

Trash

EXAM_BACKEND
10 requests

POST add movies

PATC edit movies

DEL delete movies

GET show all movies

POST add categories

PATC edit categories

DEL delete categories

GET show categories

POST add movcat

DEL delete movcat

jc8expressmysql
12 requests

Project Akhir
12 requests

ReactMongoose
13 requests

GET lo POST PATC edit DEL delete GET show POST add PATC edit DEL d GET show POST add DEL delete +

delete categories

Examples (0)

DELETE localhost:2019/deletecategories/categoriesid Send Save

Params Authorization Headers (8) Body Pre-request Script Tests Cookies Code Comments (0)

Query Params

KEY VALUE DESCRIPTION Bulk Edit

Key Value Description

Path Variables

KEY VALUE DESCRIPTION Bulk Edit

categoriesid 4 Description

Body Cookies Headers (7) Test Results Status: 200 OK Time: 372 ms Size: 324 B Save Download

Pretty Raw Preview JSON

```
1- [
2-   {
3-     "id": 1,
4-     "nama": "Korean Film"
5-   },
6-   {
7-     "id": 2,
8-     "nama": "Drama"
9-   },
10-  {
11-    "id": 3,
12-    "nama": "Comedy"
13-  }
14- ]
```

Bootcamp Build Browse

SHOW ALL CATEGORIES

The image shows the Postman application interface. On the left, the 'Collections' sidebar is open, showing a collection named 'EXAM_BACKEND' with 10 requests. The request 'show categories' is selected. The main panel shows the details of this GET request to 'localhost:2019/showcategories'. The 'Body' tab is active, displaying a JSON response in 'Pretty' format. The response is an array of 5 objects, each representing a movie category with 'id' and 'nama' fields. The status bar at the bottom indicates a successful response with status 200 OK, time 5 ms, and size 345 B.

GET localhost:2019/showcategories

Send Save

Params Authorization Headers (7) Body Pre-request Script Tests Cookies Code Comments (0)

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (7) Test Results Status: 200 OK Time: 5 ms Size: 345 B Save Download

Pretty Raw Preview JSON

```
1 - [
2   {
3     "id": 1,
4     "nama": "Sci-fi"
5   },
6   {
7     "id": 2,
8     "nama": "Drama"
9   },
10  {
11    "id": 3,
12    "nama": "Comedy"
13  },
14  {
15    "id": 4,
16    "nama": "Action"
17  },
18 ]
```

Bootcamp Build Browse

ADD MOVCAT

The screenshot shows the Postman application interface. On the left, a sidebar lists collections: EXAM_BACKEND (10 requests), jc8expressmysql (12 requests), Project Akhir (12 requests), and ReactMongoose (13 requests). The 'add movcat' request is selected under EXAM_BACKEND. The main panel shows the request details for 'add movcat' at 'localhost:2019/addmovcat'. The request is a POST method with a body of JSON data. The body is displayed in a 'Pretty' view, showing an array of three objects. The status bar at the bottom indicates 'Status: 200 OK', 'Time: 1118 ms', and 'Size: 658 B'.

POST localhost:2019/addmovcat

Params Authorization Headers (9) Body Pre-request Script Tests Cookies Code Comments (0)

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

```
1- {
2-   "movie_id": "2",
3-   "category_id": "3"
4- }
```

Body Cookies Headers (7) Test Results Status: 200 OK Time: 1118 ms Size: 658 B Save Download

Pretty Raw Preview JSON

```
1- [
2-   {
3-     "movcat_id": 5,
4-     "nama_file": "Avengers: End Game",
5-     "category": "Action"
6-   },
7-   {
8-     "movcat_id": 6,
9-     "nama_file": "Avengers: End Game",
10-    "category": "Drama"
11-  },
12-  {
13-    "movcat_id": 7,
14-    "nama_file": "Dora the Explorer",
15-    "category": "Comedy"
16-  }
17- ]
```

Bootcamp Build Browse

DELETE MOVCAT

The image shows the Postman application interface. On the left, a sidebar lists collections: EXAM_BACKEND (10 requests), jc8expressmysql (12 requests), Project Akhir (12 requests), and ReactMongoose (13 requests). The 'delete movcat' request is selected under EXAM_BACKEND. The main panel shows the request details for 'delete movcat' with a DELETE method and URL 'localhost:2019/deletemovcat/movcatid'. The 'Params' tab is active, showing a single path variable 'movcatid' with the value '7'. The 'Body' tab is also active, displaying a JSON array of movie objects. The status bar at the bottom indicates a successful 200 OK response with a time of 38 ms and a size of 590 B.

DELETE localhost:2019/deletemovcat/movcatid

Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Path Variables

KEY	VALUE	DESCRIPTION
movcatid	7	Description

Body

```
1- [
2- {
3-   "movcat_id": 5,
4-   "nama_film": "Avengers: End Game",
5-   "category": "Action"
6- },
7- {
8-   "movcat_id": 6,
9-   "nama_film": "Avengers: End Game",
10-  "category": "Drama"
11- },
12- {
13-   "movcat_id": 8,
14-   "nama_film": "Dora the Explorer",
15-   "category": "Action"
16- },
17- {
18-   "movcat_id": 9,
19-   "nama_film": "Dora the Explorer"
```

Status: 200 OK Time: 38 ms Size: 590 B