

Dion Cineplex

Theatre Service

Member

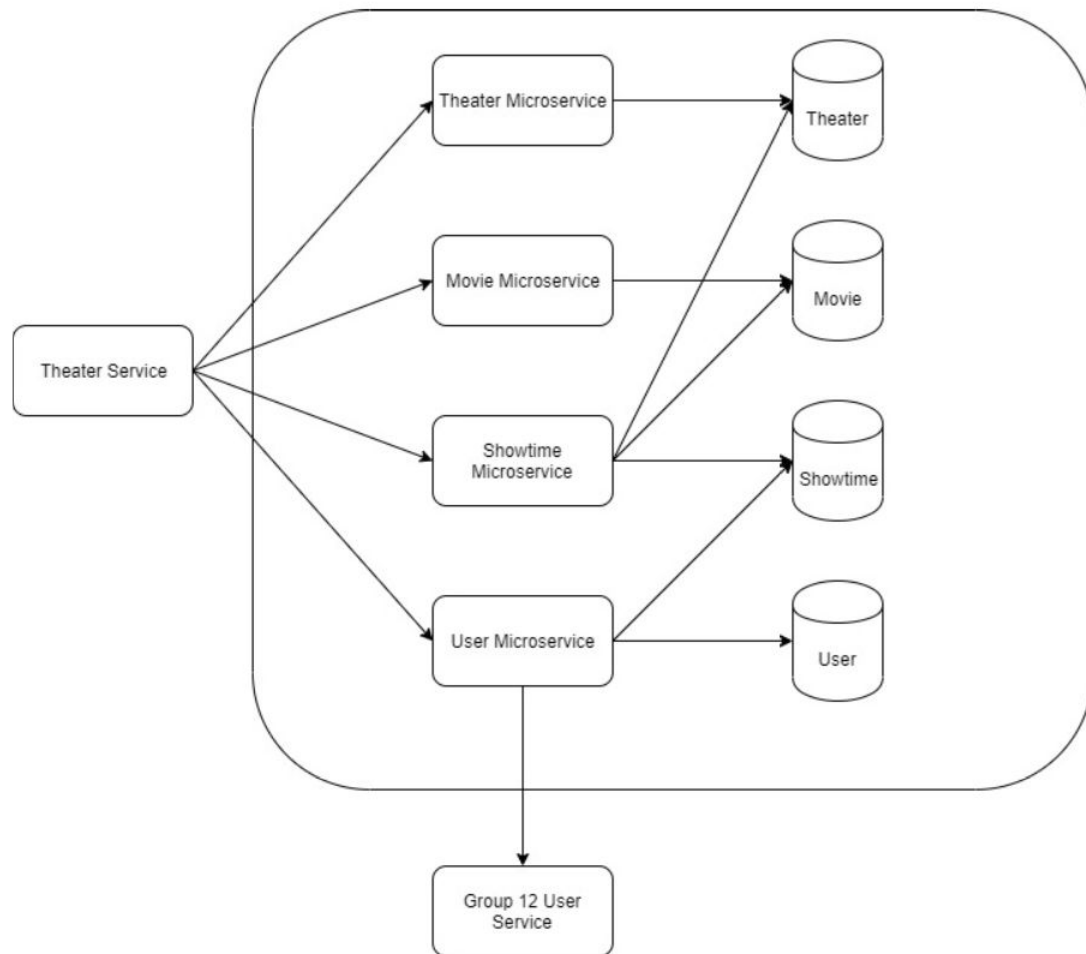
- นาย จิรพัส วงษ์พิทักษ์ 60070009
- นาย นิติ จิรการวุฒิกโร 60070041
- นาย ศุภกิตติ เรียร์ธัญญกิจ 60070098
- นาย อนุสรณ์ เม่นนาเกร็ด 60070110



What is it?

Service สำหรับระบบโรงหนังฟัง ผู้ดูแลระบบ และ ผู้ใช้งานระบบ

- ข้อมูลโรงภาพยนตร์
- ข้อมูลภาพยนตร์
- ข้อมูลรอบฉายของภาพยนตร์
- การจองที่นั่งของโรงภาพยนตร์
- ข้อมูลประวัติการซื้อตั๋ว



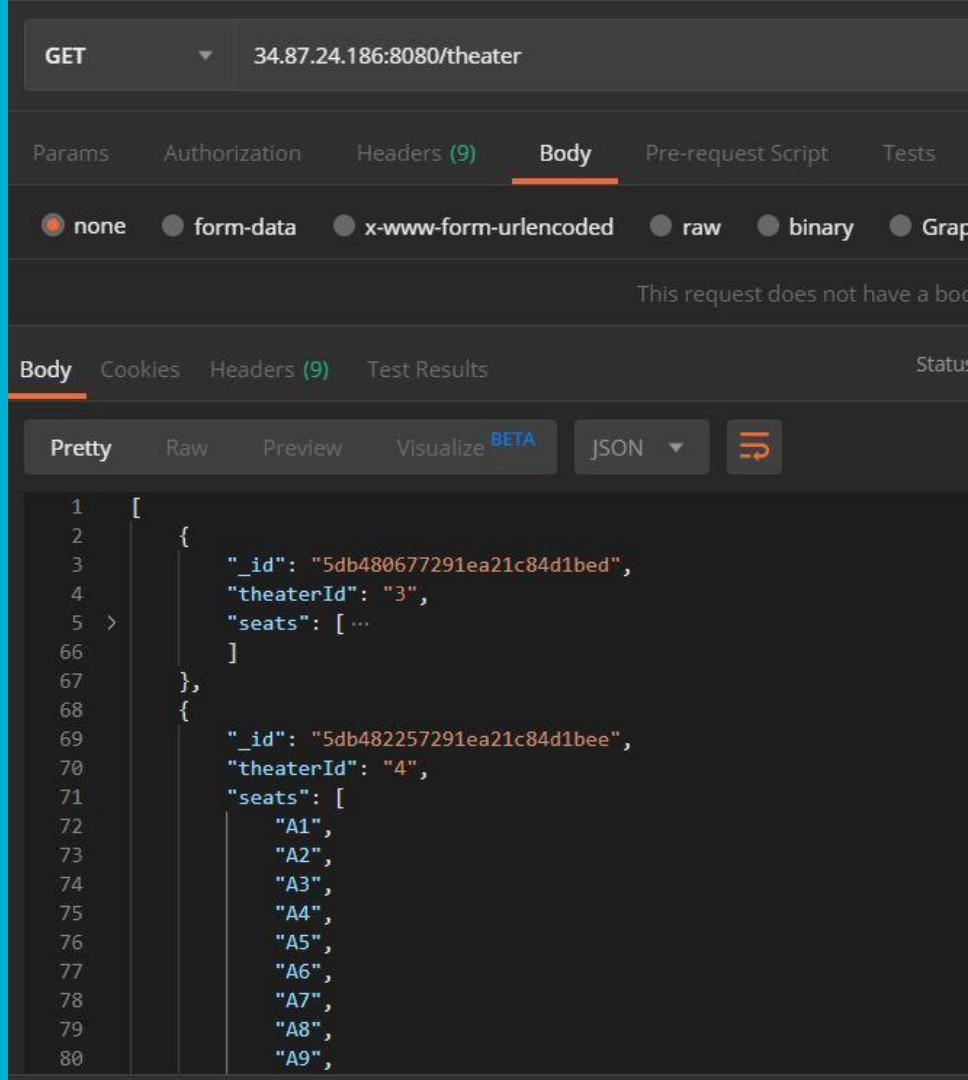
Theater

Get Theater

ใช้ดูว่าในระบบมีโรงหนังอะไรบ้าง แต่ละโรงมีที่นั่งอะไรบ้าง

URL : /theater

Method : GET

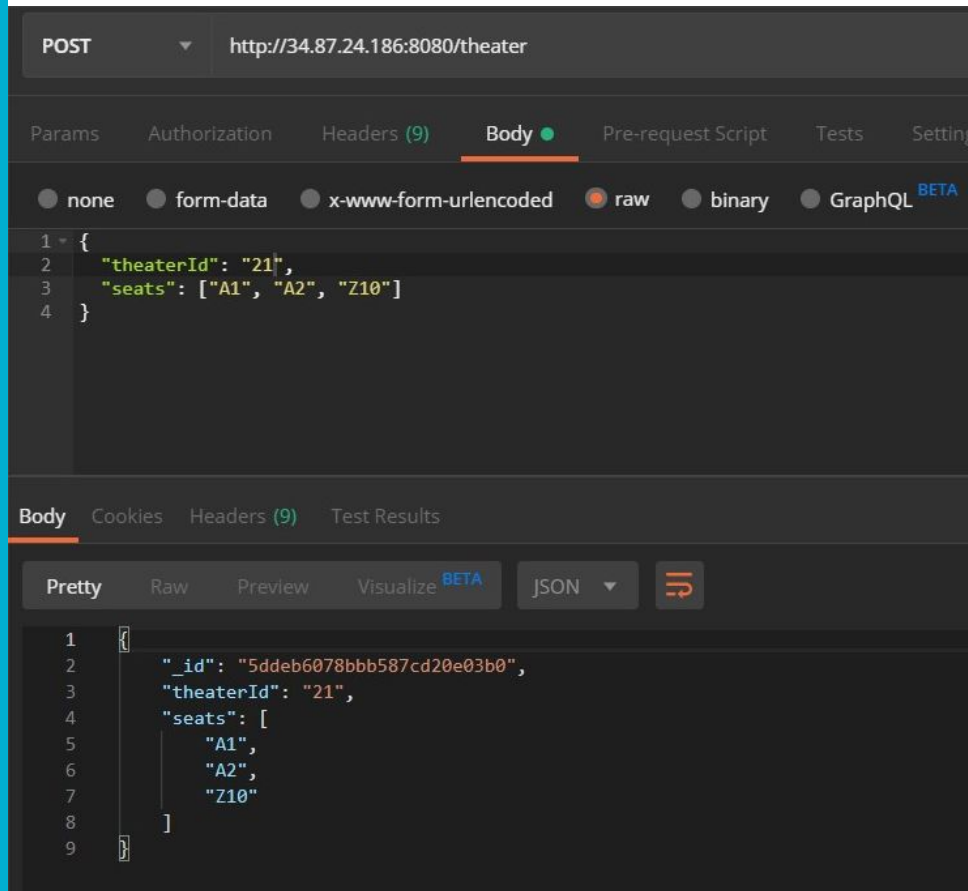


Add Theater

ใช้สำหรับการเพิ่มโรงภาพยนตร์เข้าไปในระบบ

URL : /theater

Method : POST

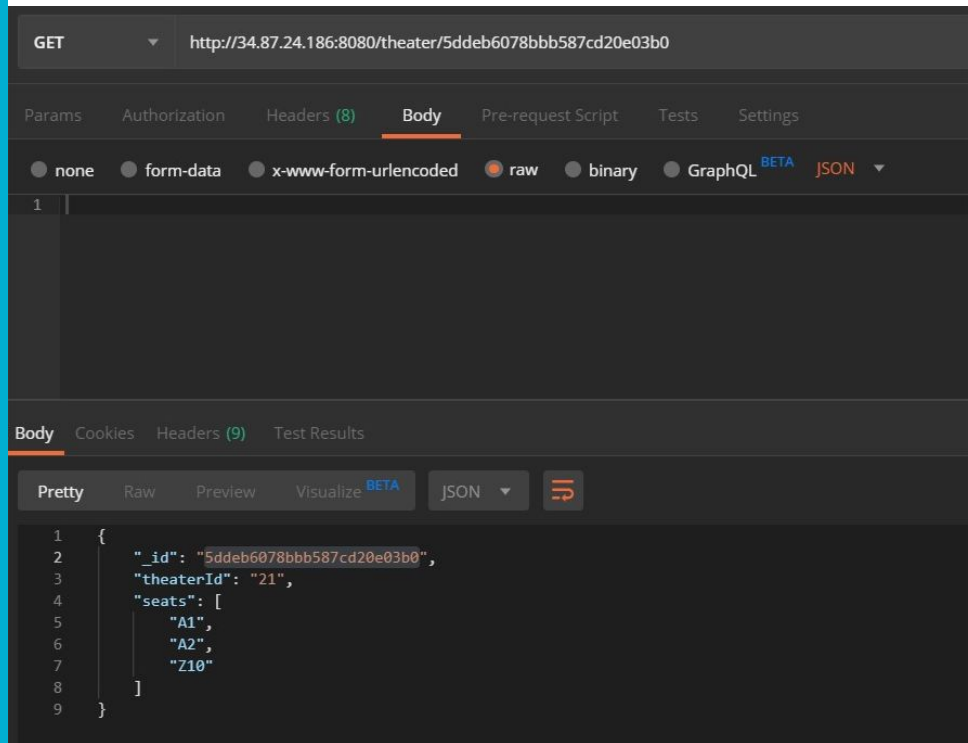


Get Theater Detail by Id

ใช้สำหรับดูข้อมูลของโรงภาพยนตร์โดยใช้ _id เป็นตัวอ้างอิง เพื่อใช้ในขั้นตอนการเพิ่มรอบฉาย

URL : /theater/{_id}

Method : GET

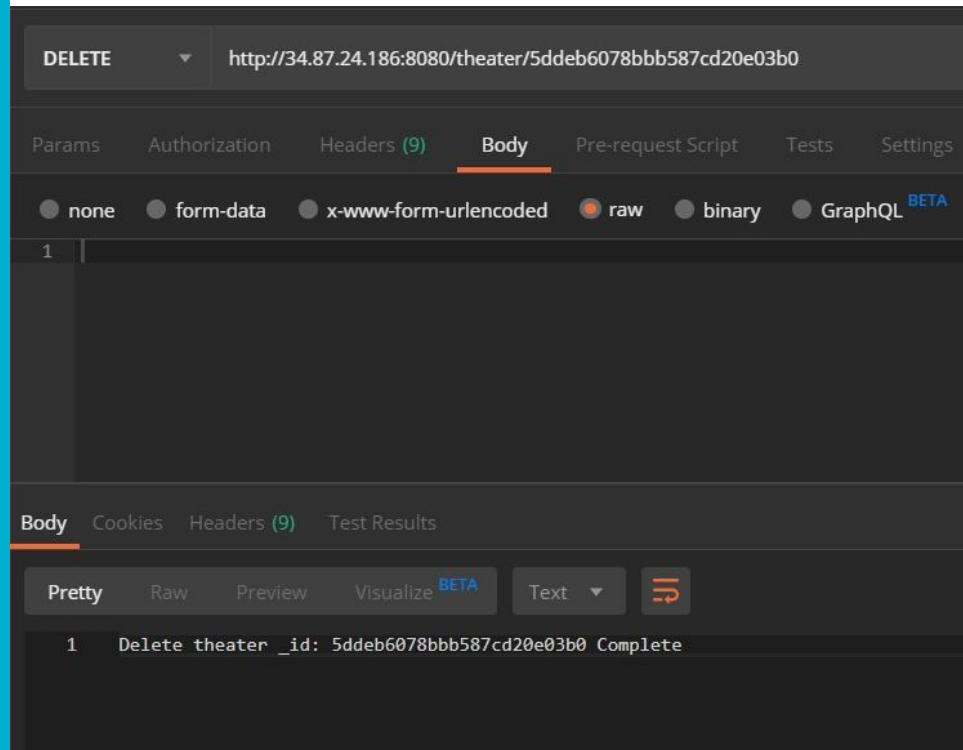


Delete Theater

ใช้สำหรับการลบโรงภาพยนตร์ในระบบ เลือกลบโดยใช้
_id เป็นตัวอ้างอิง

URL : /theater/{_id}

Method : DELETE



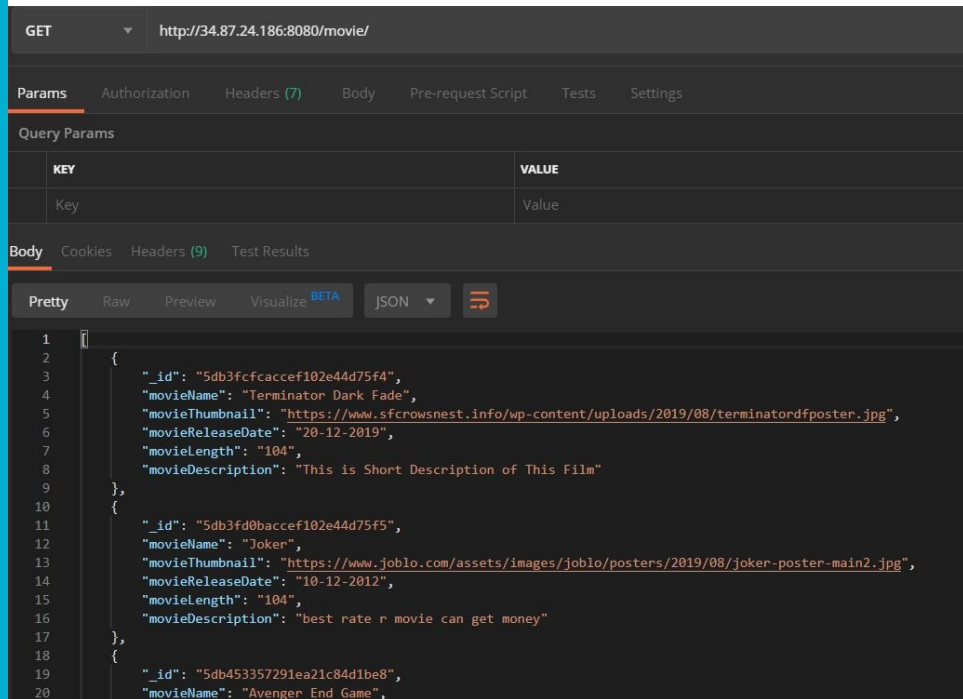
Movie

Get all Movies Information

ใช้สำหรับการเรียกดูข้อมูลของภาพยนตร์ทั้งหมดในระบบ

URL : /movie

Method : GET



The screenshot displays a REST client interface with the following details:

- Method:** GET
- URL:** http://34.87.24.186:8080/movie/
- Params:** Authorization, Headers (7), Body, Pre-request Script, Tests, Settings
- Query Params:** A table with one entry: Key (KEY) and Value (VALUE).
- Body:** Cookies, Headers (9), Test Results
- Response Format:** Pretty (JSON)
- Response Body:** A JSON array of movie objects.

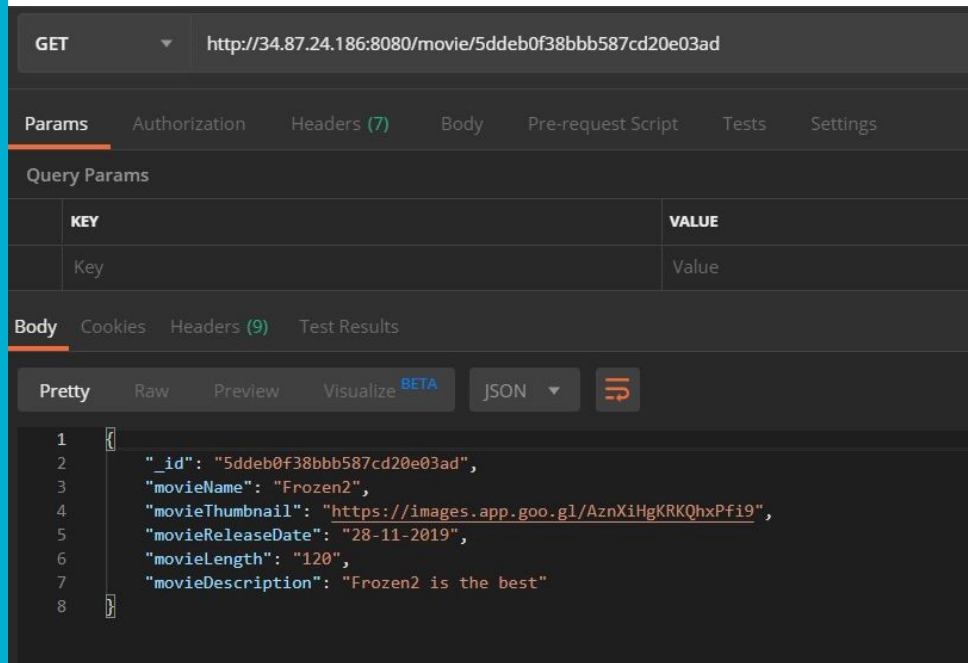
```
1 {
2   {
3     "_id": "5db3fcfcaccef102e44d75f4",
4     "movieName": "Terminator Dark Fade",
5     "movieThumbnail": "https://www.sfcrownsnest.info/wp-content/uploads/2019/08/terminatordfposter.jpg",
6     "movieReleaseDate": "20-12-2019",
7     "movieLength": "104",
8     "movieDescription": "This is Short Description of This Film"
9   },
10  {
11    "_id": "5db3fd0baccef102e44d75f5",
12    "movieName": "Joker",
13    "movieThumbnail": "https://www.joblo.com/assets/images/joblo/posters/2019/08/joker-poster-main2.jpg",
14    "movieReleaseDate": "10-12-2012",
15    "movieLength": "104",
16    "movieDescription": "best rate r movie can get money"
17  },
18  {
19    "_id": "5db453357291ea21c84d1be8",
20    "movieName": "Avenger End Game",
```

Get all Movies Information by Id

ใช้สำหรับดูข้อมูลของภาพยนตร์โดยใช้ _id
เป็นตัวอ้างอิง เพื่อใช้ในขั้นตอนการเพิ่มรอบฉาย

URL : /movie/{_id}

Method : GET



The screenshot shows a REST client interface with a GET request to the URL `http://34.87.24.186:8080/movie/5ddebf38bbb587cd20e03ad`. The response is a JSON object with the following fields:

KEY	VALUE
Key	Value

The response body is displayed in a code editor with line numbers 1 through 8. The JSON data is as follows:

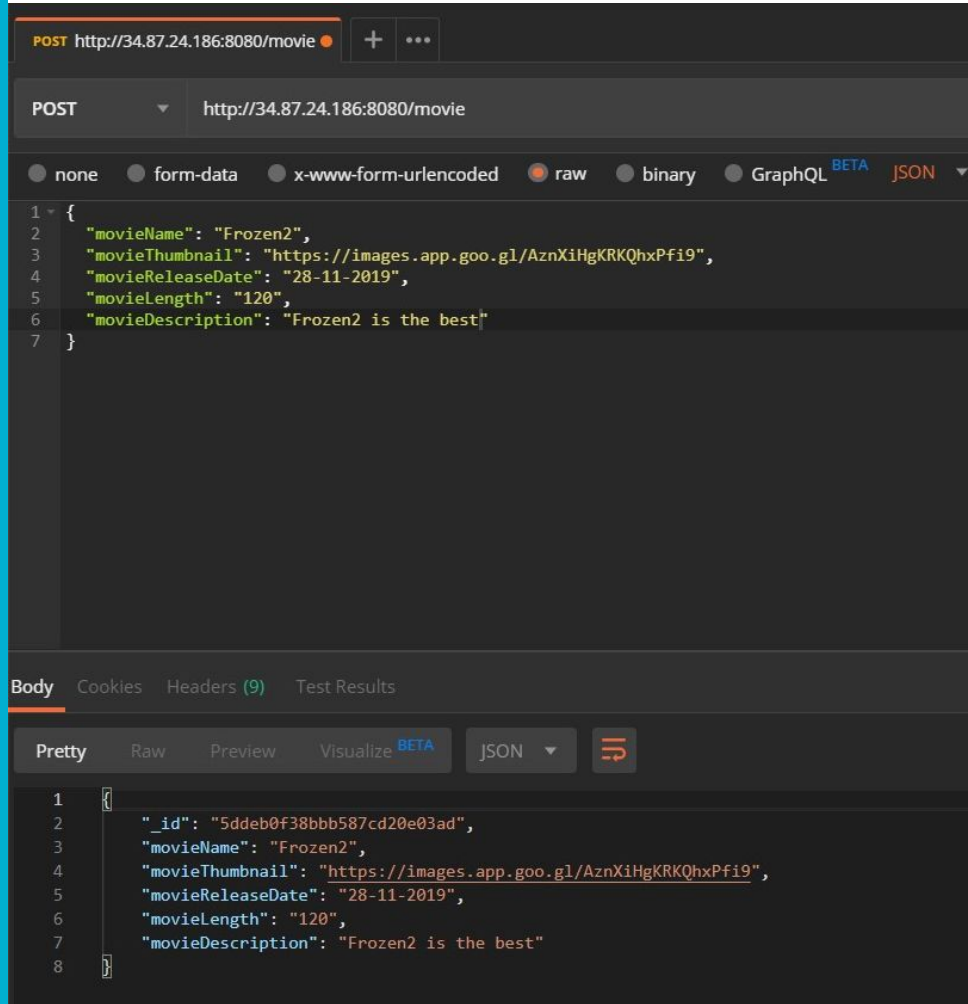
```
1 {  
2   "_id": "5ddebf38bbb587cd20e03ad",  
3   "movieName": "Frozen2",  
4   "movieThumbnail": "https://images.app.goo.gl/AznXiHgKRKQhxFi9",  
5   "movieReleaseDate": "28-11-2019",  
6   "movieLength": "120",  
7   "movieDescription": "Frozen2 is the best"  
8 }
```

Add Movie

ใช้สำหรับการเพิ่มข้อมูลภาพยนตร์เข้าไปในระบบ

URL : /movie

Method : POST

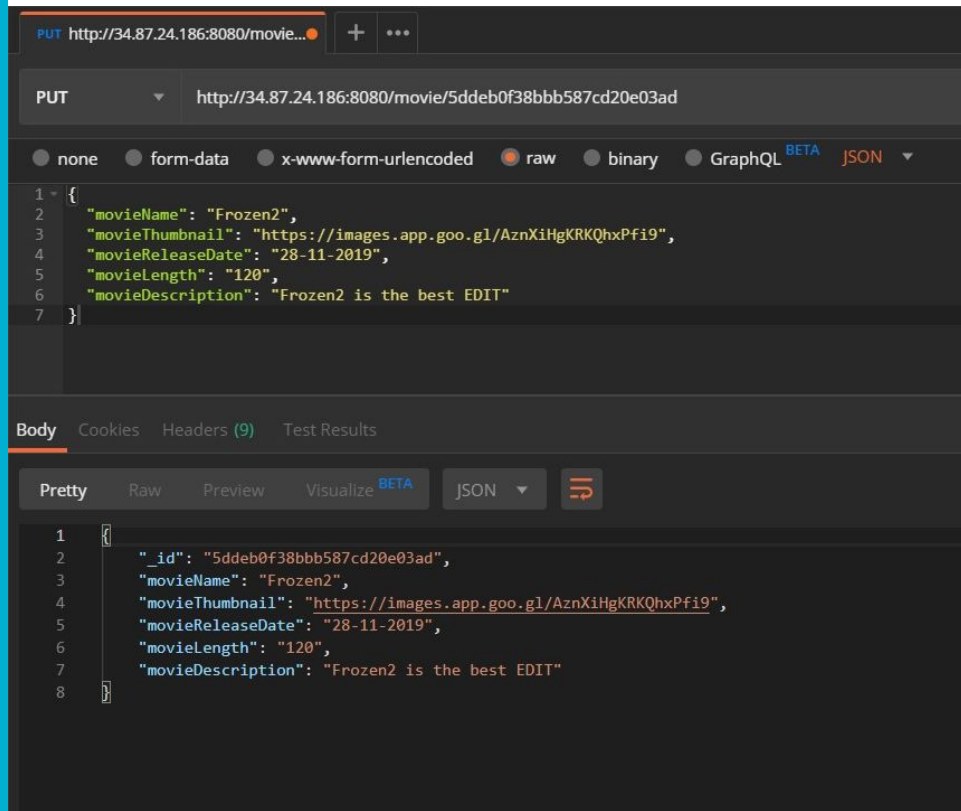


Edit Movie

ใช้สำหรับการแก้ไขข้อมูลของภาพยนตร์โดยใช้
_id เป็นตัวอ้างอิง

URL : /movie/{_id}

Method : PUT

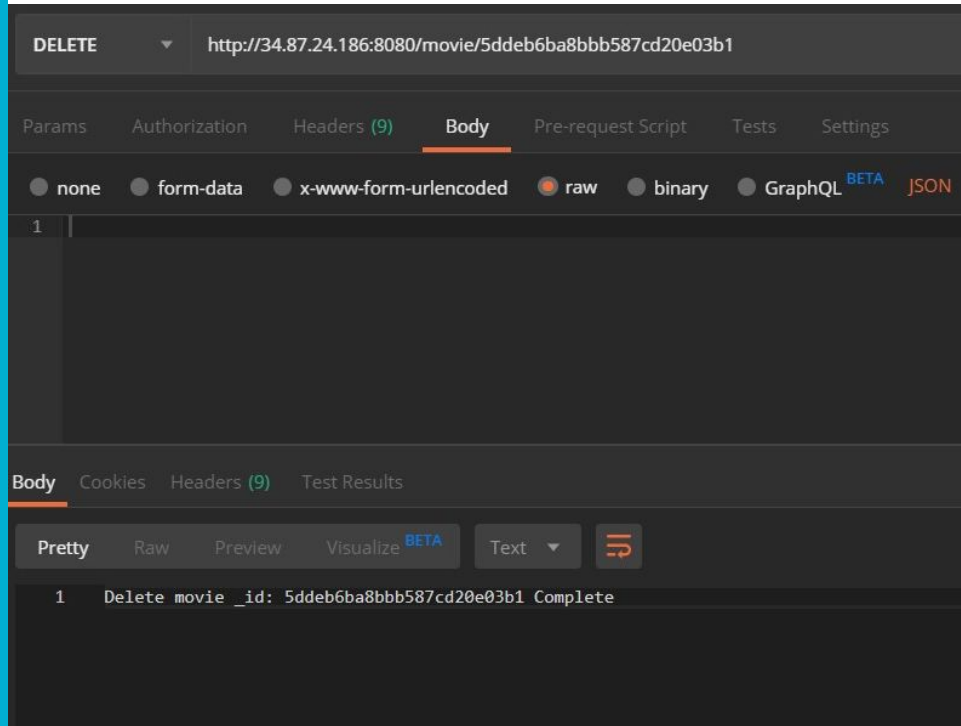


Delete Movie

ใช้สำหรับการลบข้อมูลของภาพยนตร์โดยใช้
_id เป็นตัวอ้างอิง

URL : /movie/{_id}

Method : DELETE



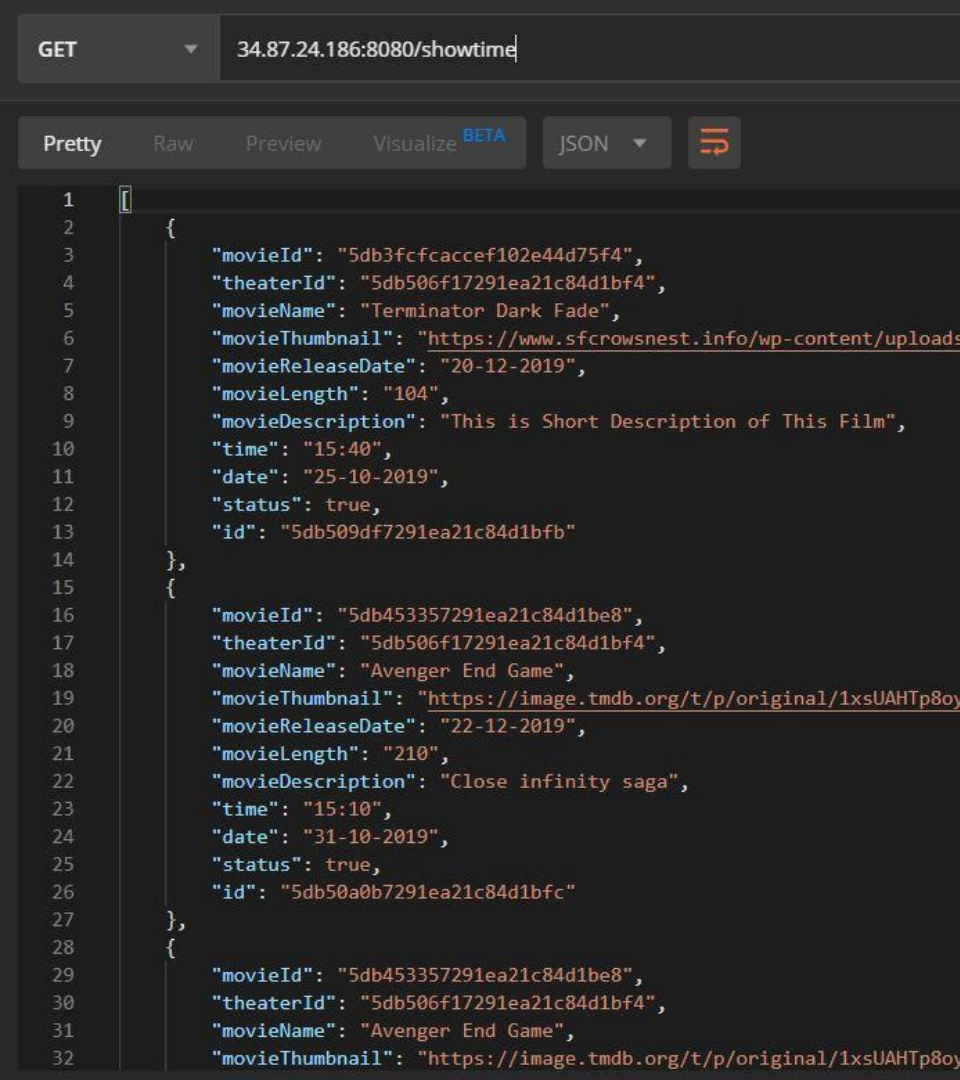
Showtime

Get all Showtimes

ใช้สำหรับการเรียกดูข้อมูลของรอบฉายทั้งหมดในระบบ

URL : /showtime

Method : GET

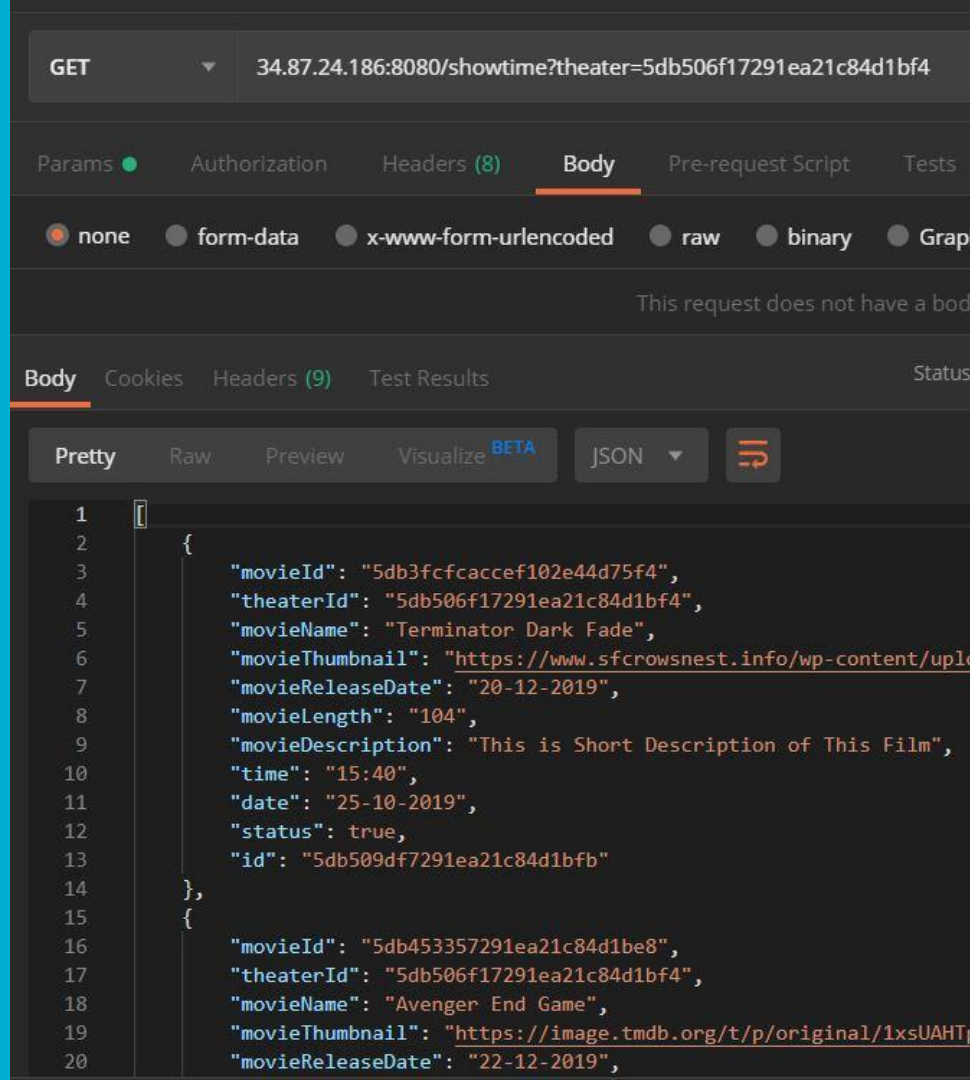


Get all Showtimes by TheaterId

ใช้สำหรับการเรียกดูข้อมูลของรอบฉายทั้งหมดในระบบ โดยใช้ theaterId เป็นตัวอ้างอิง

URL :
/showtime?theater={theaterId}

Method : GET

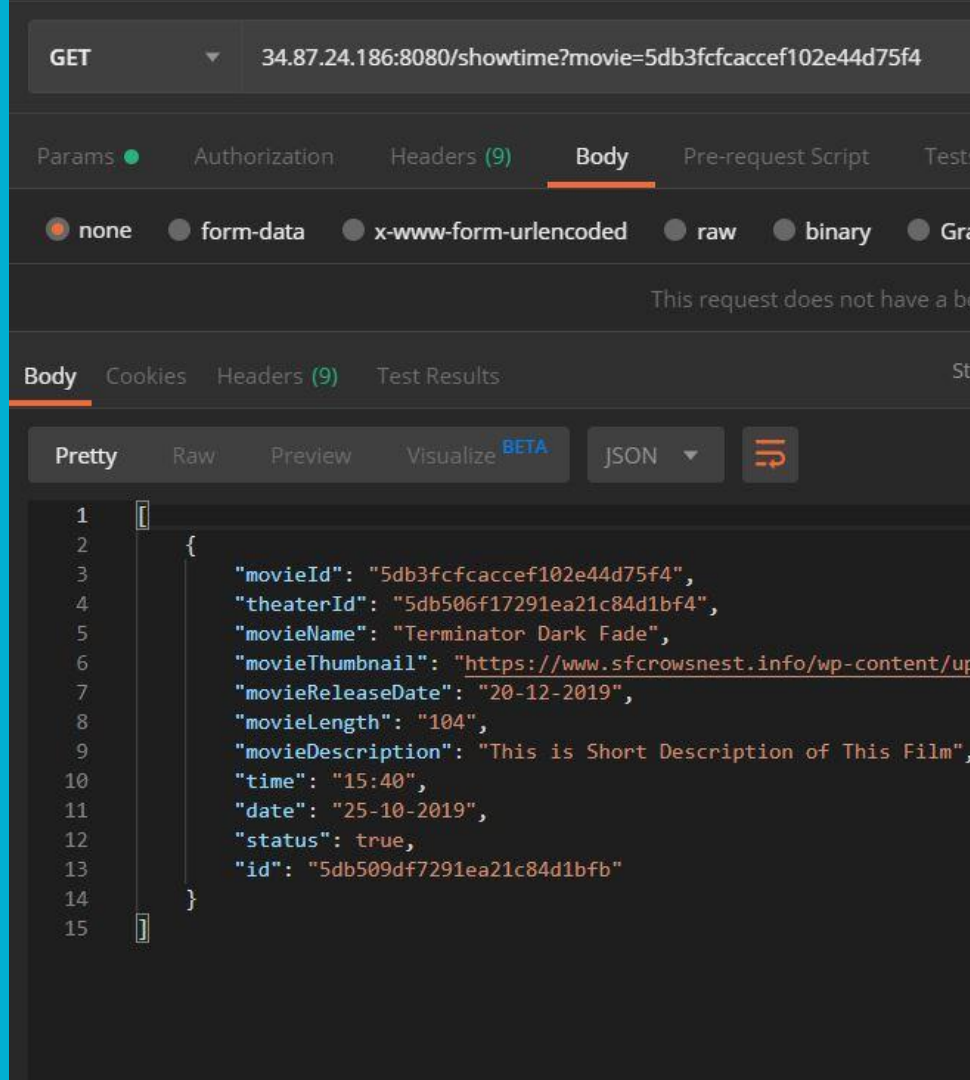


Get all Showtimes by MovieId

ใช้สำหรับการเรียกดูข้อมูลของรอบฉายทั้งหมดในระบบ โดยใช้ movieId เป็นตัวอ้างอิง

URL:
/showtime?movie={movieId}

Method : GET



Get all Showtimes by Id

ใช้สำหรับการเรียกดูข้อมูลของรอบฉายทั้งหมดในระบบ โดยใช้ _id เป็นตัวอ้างอิง

URL:
/showtime?id={_id}

Method : GET

GET

34.87.24.186:8080/showtime?id=5db509c37291ea21c84d1bf9

Pretty

Raw

Preview

Visualize BETA

JSON



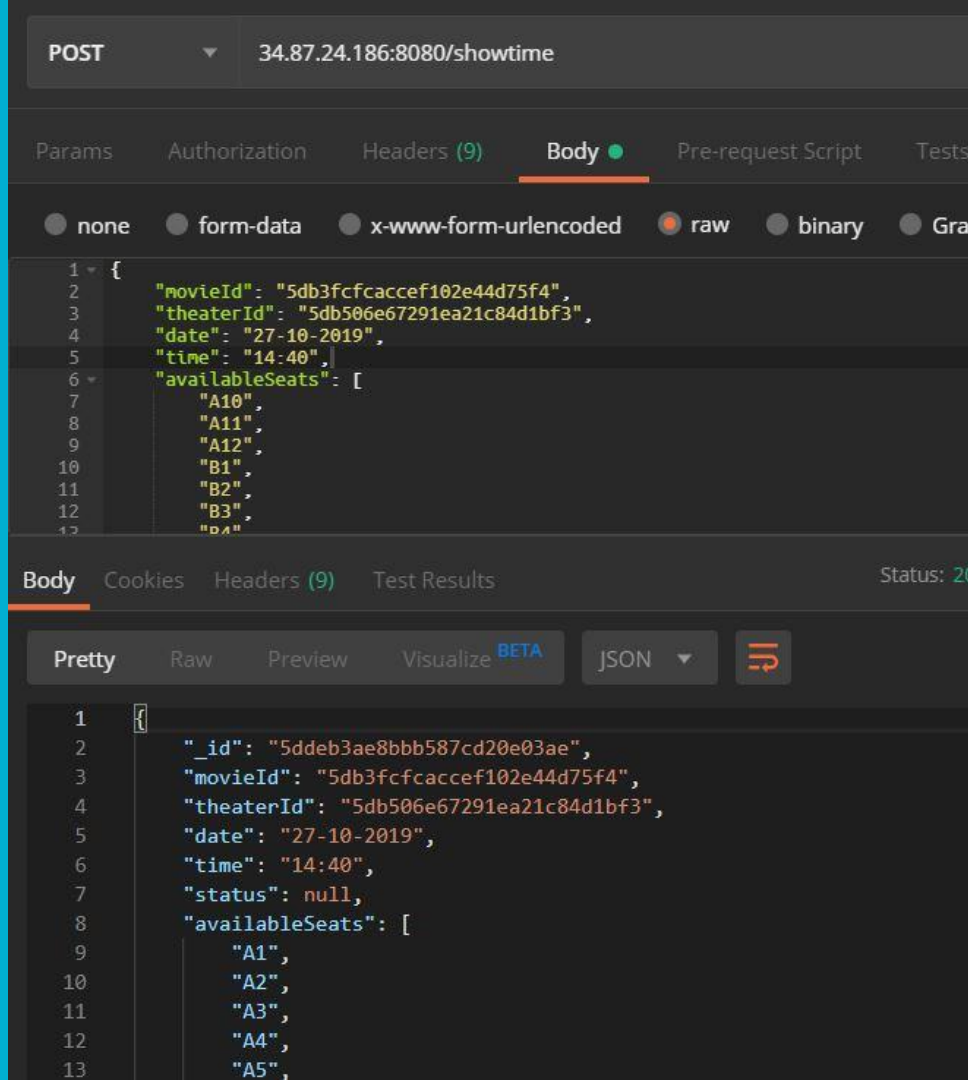
```
1 {
2   "_id": "5db509c37291ea21c84d1bf9",
3   "movieId": "5db3fcfcaccef102e44d75f4",
4   "theaterId": "5db506e67291ea21c84d1bf3",
5   "date": "29-10-2019",
6   "time": "14:40",
7   "status": true,
8   "availableSeats": [
9     "A10",
10    "A11",
11    "A12",
12    "B1",
13    "B2",
14    "B3",
15    "B4",
16    "B5",
17    "B6",
18    "B7",
19    "B8",
20    "B9",
21    "B10",
22    "B11",
23    "B12",
24    "C1",
25    "C2",
26    "C3",
27    "C4",
28    "C5",
29    "C6",
30    "C7",
31    "C8",
```

Add Showtime

ใช้สำหรับการเพิ่มรอบฉาย

URL : /showtime

Method : POST

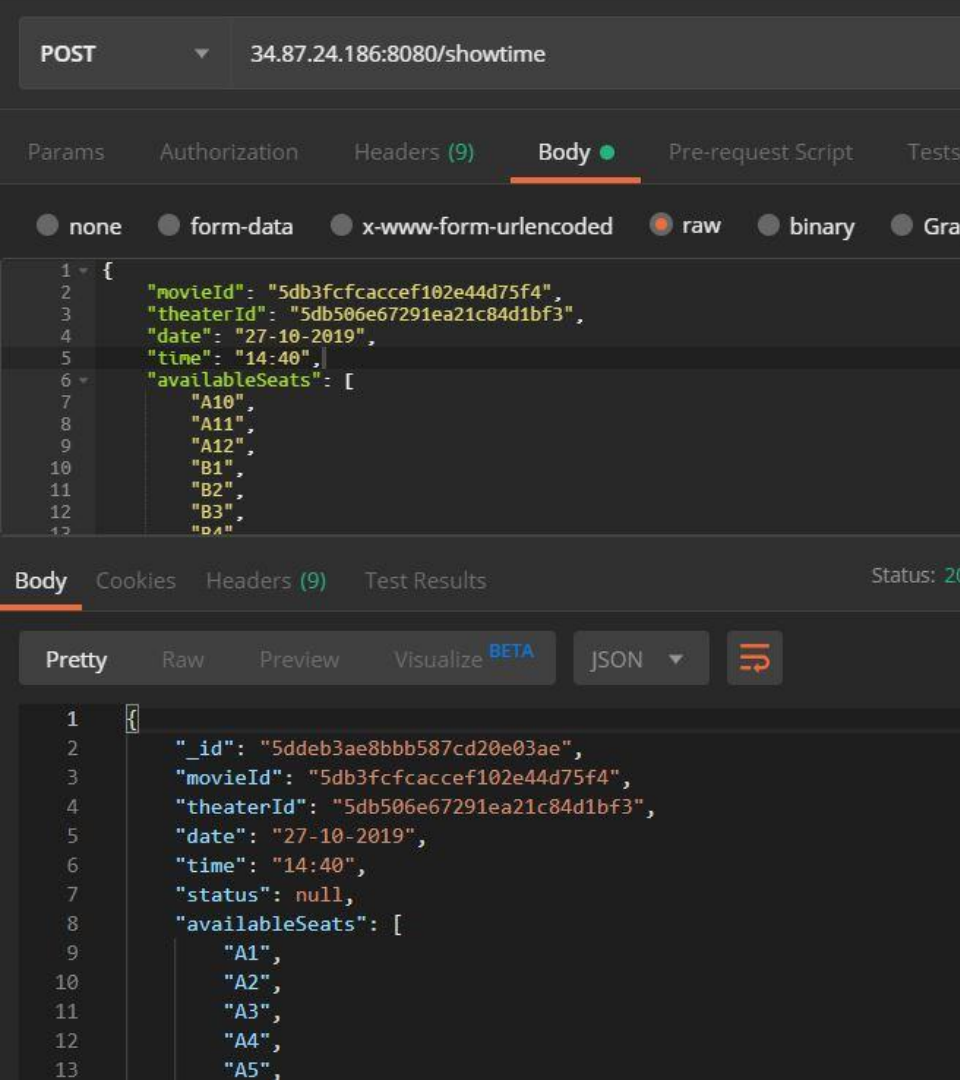


Edit Showtime

ใช้สำหรับการจอง ที่จะมีการปรับข้อมูลที่นั่งที่ยังสามารถจองได้ให้ลดลงเป็นหลัก และสามารถแก้ไขข้อมูลอื่นของรอบฉายได้

URL : /showtime/{_id}

Method : PUT

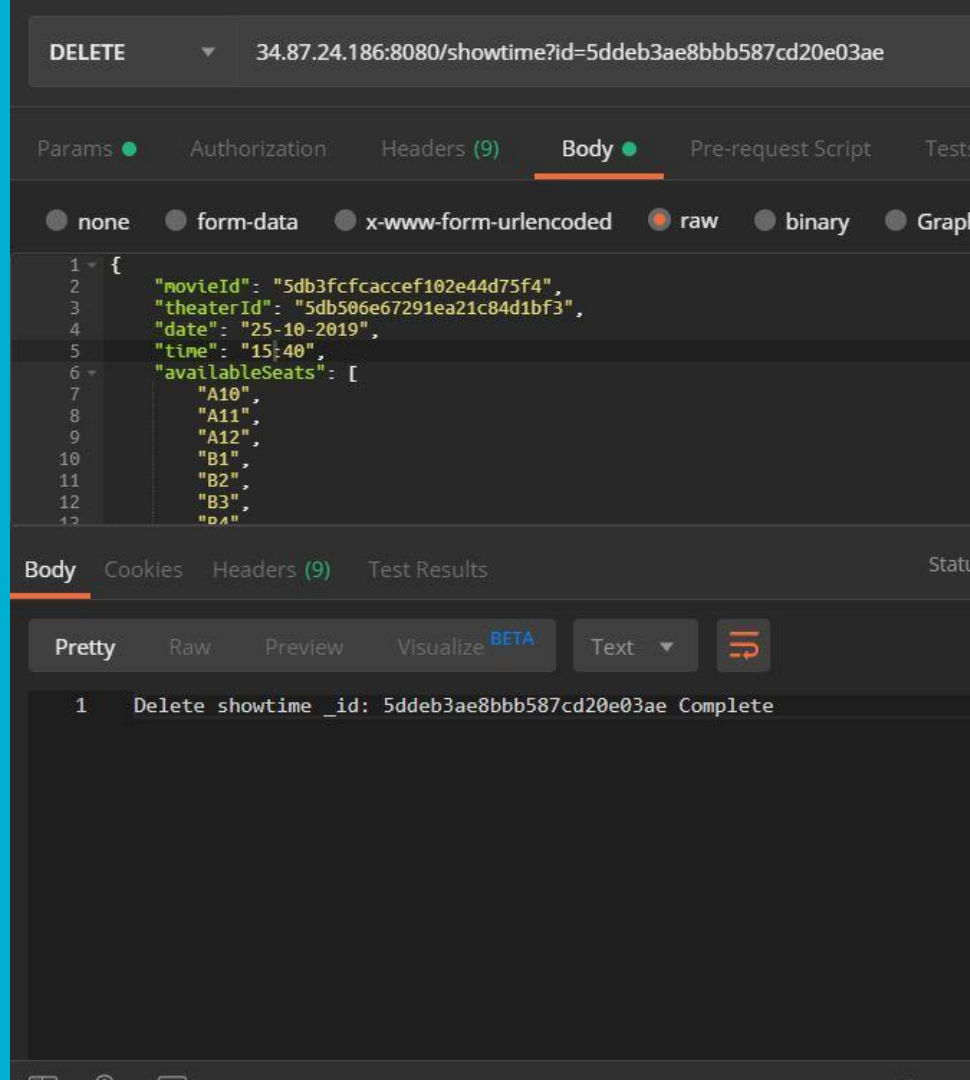


Delete Showtime by Id

ใช้สำหรับการลบรอบฉายโดยใช้ _id เป็นตัวอ้างอิง

URL : /showtime?id={_id}

Method : DELETE

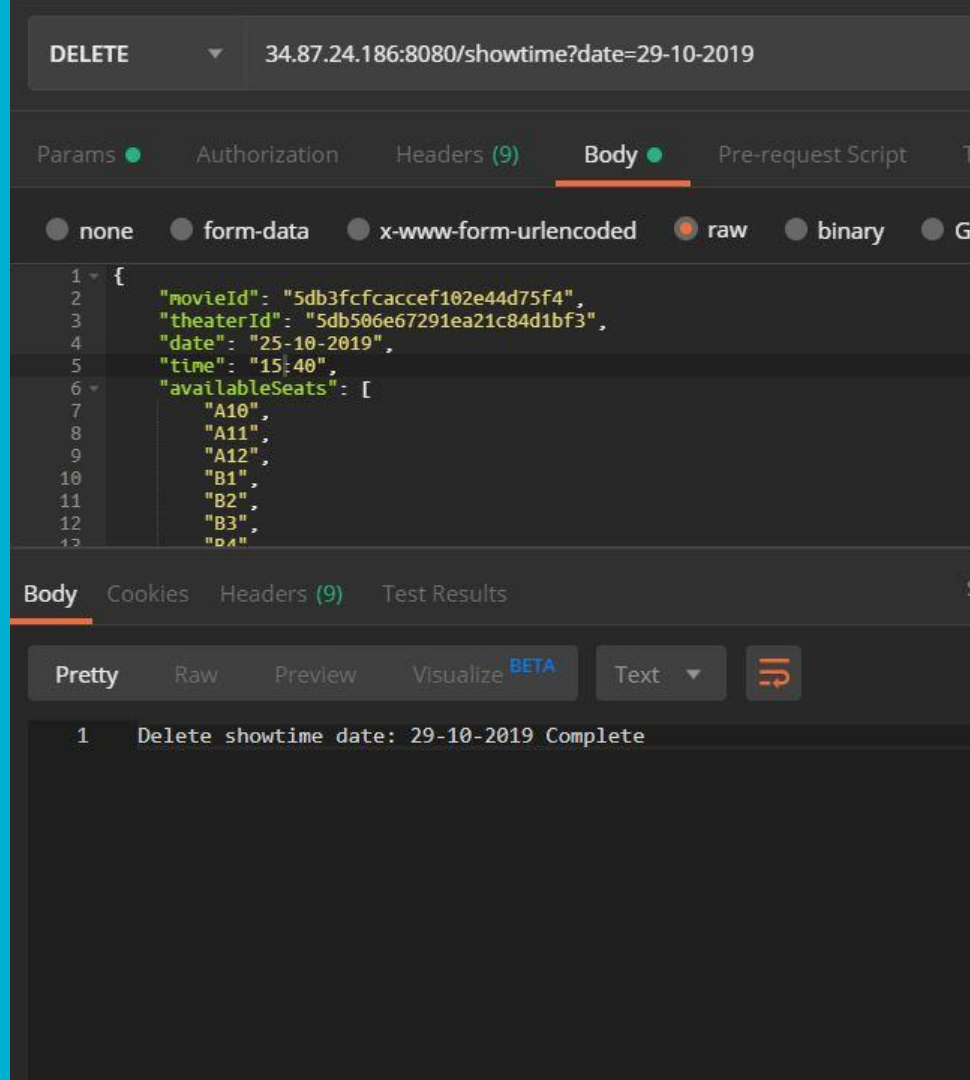


Delete Showtime by Date

ใช้สำหรับการลบรอบฉายโดยใช้ date เป็นตัวอ้างอิง
สำหรับการใช้ลบรอบฉายเมื่อหมดวัน

URL : /showtime?date={date}

Method : DELETE



User Utility

User Utility

User Utility

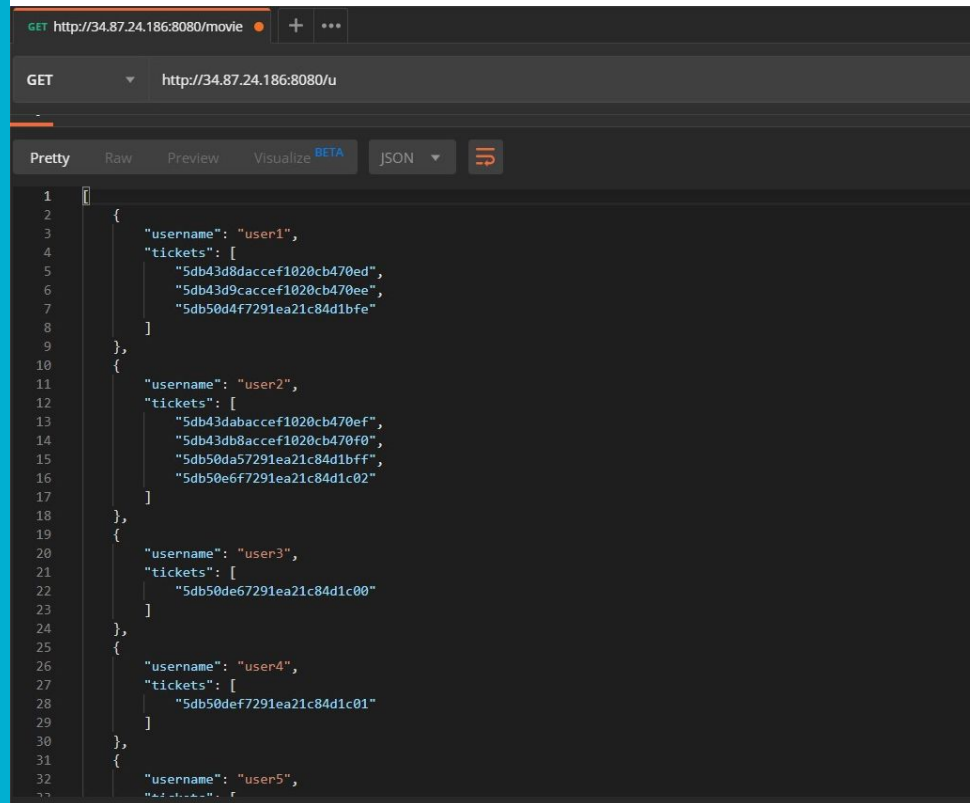
- ใช้งาน Service การ Login ของกลุ่มที่ 12
- มีเพียงการเก็บข้อมูลที่เกี่ยวข้องกับประวัติการซื้อของผู้ใช้เท่านั้น
- สิ่งที่ใช้อ้างอิงผู้ใช้คือ username
- ข้อมูลอื่นๆของผู้ใช้จะอยู่ใน Service ที่แยกกัน

Get Users Information

ใช้สำหรับการดูข้อมูลตัวของผู้ใช้ทั้งหมด

URL : /u

Method : GET



The screenshot shows a web browser interface with a GET request to `http://34.87.24.186:8080/u`. The response is displayed in JSON format, showing a list of five users. Each user object contains a `username` and an array of `tickets` (represented by UUID strings).

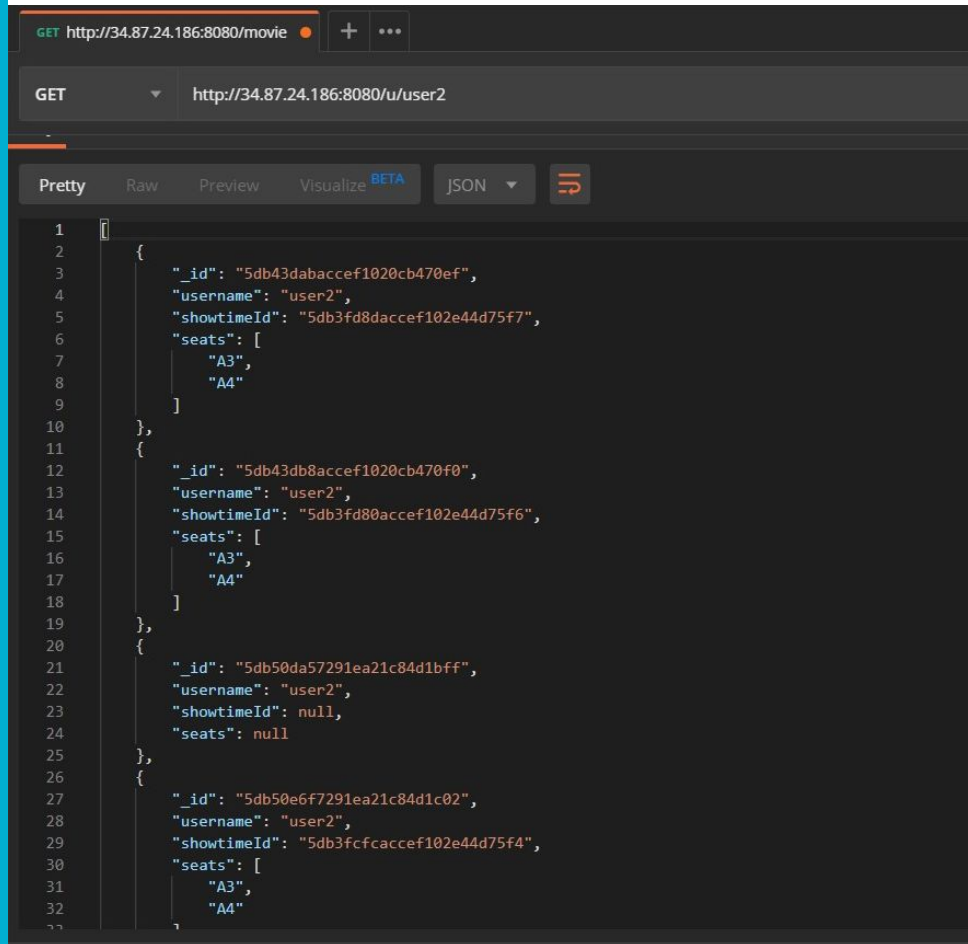
```
1 {
2   {
3     "username": "user1",
4     "tickets": [
5       "5db43d8daccef1020cb470ed",
6       "5db43d9caccef1020cb470ee",
7       "5db50d4f7291ea21c84d1bfe"
8     ]
9   },
10  {
11    "username": "user2",
12    "tickets": [
13      "5db43dabaccef1020cb470ef",
14      "5db43db8accef1020cb470f0",
15      "5db50da57291ea21c84d1bff",
16      "5db50e6f7291ea21c84d1c02"
17    ]
18  },
19  {
20    "username": "user3",
21    "tickets": [
22      "5db50de67291ea21c84d1c00"
23    ]
24  },
25  {
26    "username": "user4",
27    "tickets": [
28      "5db50def7291ea21c84d1c01"
29    ]
30  },
31  {
32    "username": "user5",
33    "tickets": [
```

Get User's Tickets

ใช้สำหรับการดูข้อมูลตั๋วทั้งหมดของผู้ใช้ โดยมี username เป็นตัวอ้างอิง

URL : /u/username

Method : GET



The screenshot shows a web browser interface with a dark theme. The address bar displays a GET request to `http://34.87.24.186:8080/u/user2`. Below the address bar, there are tabs for 'Pretty', 'Raw', 'Preview', 'Visualize BETA', and 'JSON'. The 'JSON' tab is selected, showing a JSON array of three objects. The first two objects represent users with tickets, and the third represents a user without tickets. The JSON is formatted with syntax highlighting and line numbers on the left.

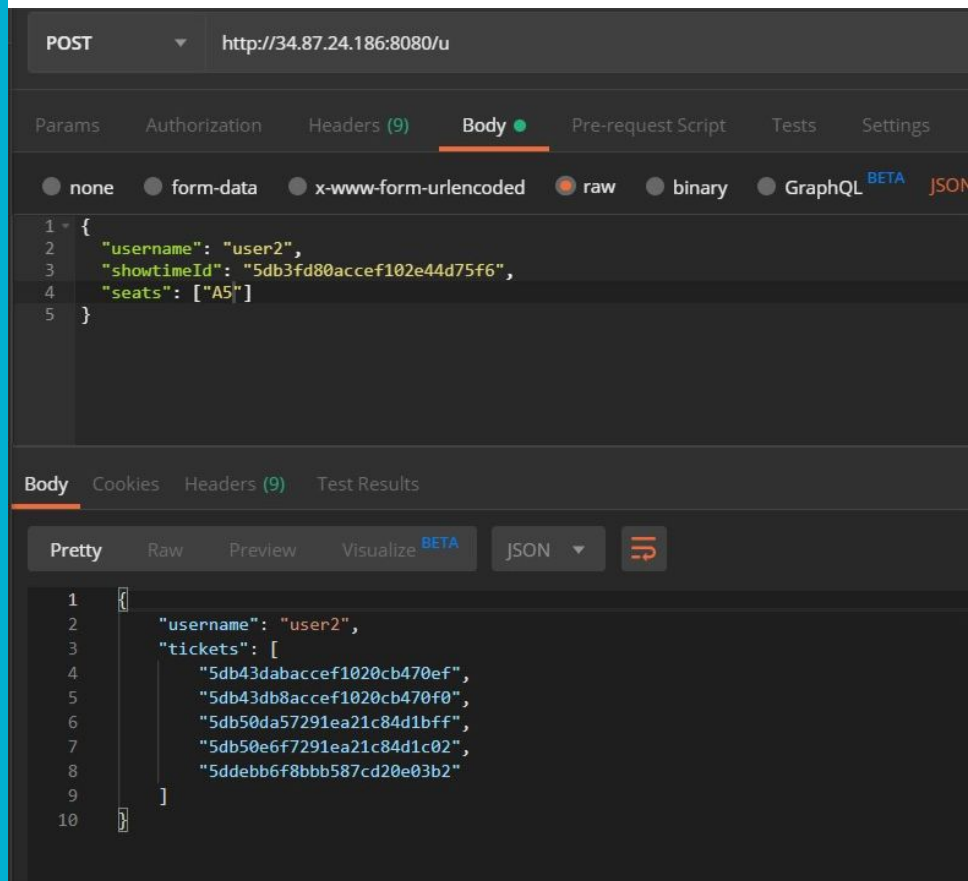
```
1  [
2    {
3      "_id": "5db43dabaccef1020cb470ef",
4      "username": "user2",
5      "showtimeId": "5db3fd8daccef102e44d75f7",
6      "seats": [
7        "A3",
8        "A4"
9      ]
10   },
11   {
12     "_id": "5db43db8accef1020cb470f0",
13     "username": "user2",
14     "showtimeId": "5db3fd80accef102e44d75f6",
15     "seats": [
16       "A3",
17       "A4"
18     ]
19   },
20   {
21     "_id": "5db50da57291ea21c84d1bff",
22     "username": "user2",
23     "showtimeId": null,
24     "seats": null
25   },
26   {
27     "_id": "5db50e6f7291ea21c84d1c02",
28     "username": "user2",
29     "showtimeId": "5db3fcfcaccef102e44d75f4",
30     "seats": [
31       "A3",
32       "A4"
33     ]
34   }
35 ]
```

Reserve

ใช้สำหรับการส่งข้อมูลการจองของผู้ใช้

URL : /u

Method : POST



Get all Tickets

ใช้สำหรับการดูข้อมูลตั๋วทั้งหมดในระบบ

URL : /u/ticket

Method : GET

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** http://34.87.24.186:8080/u/ticket
- Params:** No query parameters are present.
- Body:** The response body is displayed in JSON format, showing an array of two ticket objects. The first object has an ID of "5db43d8daccef1020cb470ed" and the second has an ID of "5db43d9caccef1020cb470ee". Both tickets are for "user1" and have two seats, "A1" and "A2".

```
1 {
2   {
3     "_id": "5db43d8daccef1020cb470ed",
4     "username": "user1",
5     "showtimeId": "5db3fd80accef102e44d75f6",
6     "seats": [
7       "A1",
8       "A2"
9     ]
10  },
11  {
12    "_id": "5db43d9caccef1020cb470ee",
13    "username": "user1",
14    "showtimeId": "5db3fd8daccef102e44d75f7",
15    "seats": [
16      "A1",
17      "A2"
18    ]
19  },
20  {
```

Delete Ticket

ใช้สำหรับการลบตั๋วในระบบ

URL : /u/ticket/{_id}

Method : DELETE

