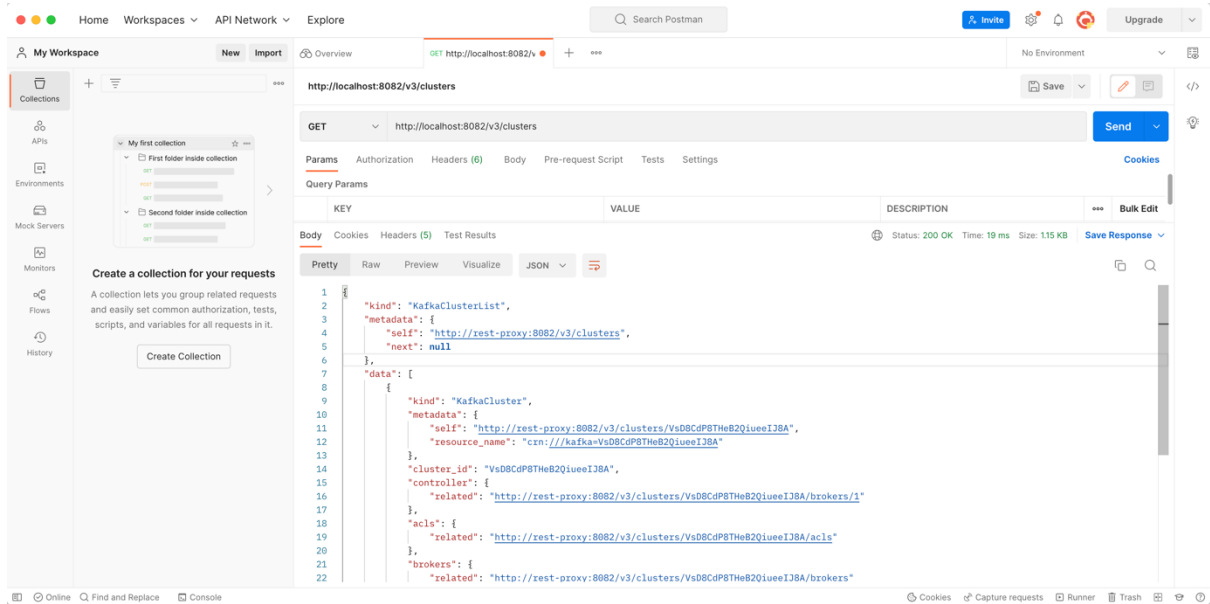


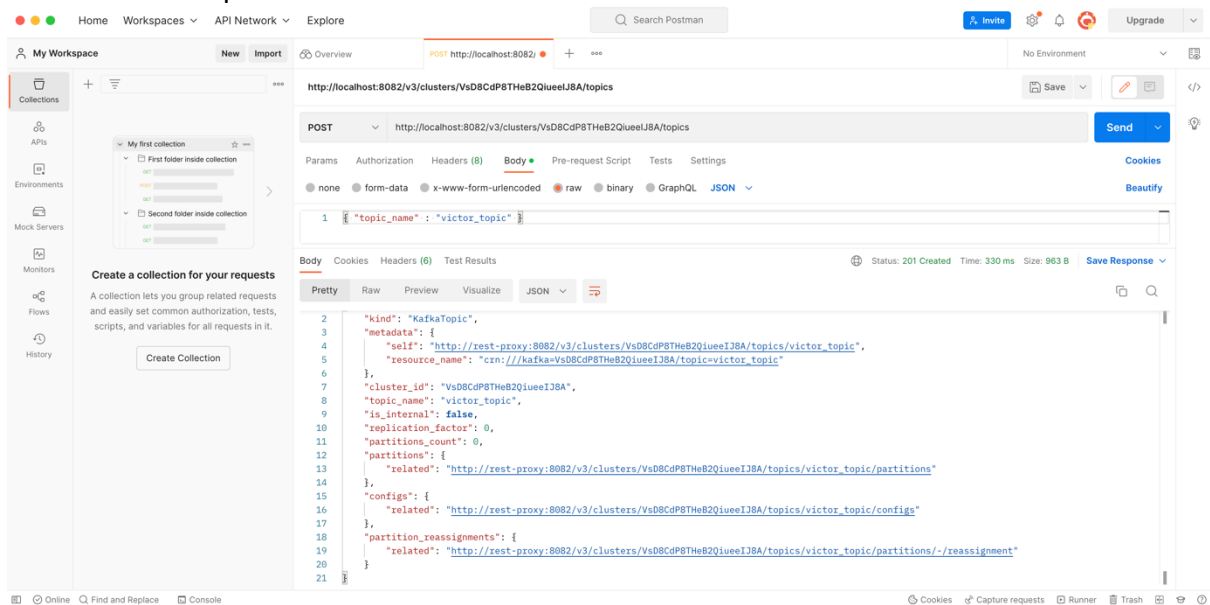
1. Get cluster ID



The screenshot shows the Postman interface with a GET request to `http://localhost:8082/v3/clusters`. The response is a JSON array of KafkaCluster objects. The first object contains the cluster_id: `VsD8CdP8TheB2QIueeI38A`.

```
1 {
2   "kind": "KafkaClusterList",
3   "metadata": {
4     "self": "http://rest-proxy:8082/v3/clusters",
5     "next": null
6   },
7   "data": [
8     {
9       "kind": "KafkaCluster",
10      "metadata": {
11        "self": "http://rest-proxy:8082/v3/clusters/VsD8CdP8TheB2QIueeI38A",
12        "resource_name": "crn:///kafka=VsD8CdP8TheB2QIueeI38A"
13      },
14      "cluster_id": "VsD8CdP8TheB2QIueeI38A",
15      "controller": {
16        "related": "http://rest-proxy:8082/v3/clusters/VsD8CdP8TheB2QIueeI38A/brokers/1"
17      },
18      "acls": {
19        "related": "http://rest-proxy:8082/v3/clusters/VsD8CdP8TheB2QIueeI38A/acls"
20      },
21      "brokers": {
22        "related": "http://rest-proxy:8082/v3/clusters/VsD8CdP8TheB2QIueeI38A/brokers"
23      }
24    }
25  ]
26 }
```

2. Create topic



The screenshot shows the Postman interface with a POST request to `http://localhost:8082/v3/clusters/VsD8CdP8TheB2QIueeI38A/topics`. The request body is a JSON object with `topic_name: 'victor_topic'`.

```
1 {
2   "topic_name": "victor_topic"
3 }
```

The response is a JSON object with the following structure:

```
1 {
2   "kind": "KafkaTopic",
3   "metadata": {
4     "self": "http://rest-proxy:8082/v3/clusters/VsD8CdP8TheB2QIueeI38A/topics/victor_topic",
5     "resource_name": "crn:///kafka=VsD8CdP8TheB2QIueeI38A/topic=victor_topic"
6   },
7   "cluster_id": "VsD8CdP8TheB2QIueeI38A",
8   "topic_name": "victor_topic",
9   "is_internal": false,
10  "replication_factor": 0,
11  "partitions_count": 0,
12  "partitions": {
13    "related": "http://rest-proxy:8082/v3/clusters/VsD8CdP8TheB2QIueeI38A/topics/victor_topic/partitions"
14  },
15  "configs": {
16    "related": "http://rest-proxy:8082/v3/clusters/VsD8CdP8TheB2QIueeI38A/topics/victor_topic/configs"
17  },
18  "partition_reassignments": {
19    "related": "http://rest-proxy:8082/v3/clusters/VsD8CdP8TheB2QIueeI38A/topics/victor_topic/partitions/-/reassignment"
20  }
21 }
```

3. List all topics

The screenshot shows the Postman interface with a GET request to `http://localhost:8082/v3/clusters/VsD8CdP8THeB2QIueeI38A/topics`. The request is successful, returning a 200 OK status. The response body is a JSON array of Kafka topics. The first topic is `"victor_topic"`.

```
1 {
2   "topic_name": "victor_topic"
3 }
```

The response body is displayed in the Pretty view, showing a JSON array of Kafka topics. The first topic is `"victor_topic"`.

```
1 {
2   "kind": "KafkaTopicList",
3   "metadata": {
4     "self": "http://rest-proxy:8082/v3/clusters/VsD8CdP8THeB2QIueeI38A/topics",
5     "next": null
6   },
7   "data": [
8     {
9       "kind": "KafkaTopic",
10      "metadata": {
11        "self": "http://rest-proxy:8082/v3/clusters/VsD8CdP8THeB2QIueeI38A/topics/_confluent-command",
12        "resource_name": "crn:///kafka-VsD8CdP8THeB2QIueeI38A/topic=_confluent-command"
13      },
14      "cluster_id": "VsD8CdP8THeB2QIueeI38A",
15      "topic_name": "_confluent-command",
16      "is_internal": false,
17      "replication_factor": 1,
18      "partitions_count": 1,
19      "partitions": {
20        "related": "http://rest-proxy:8082/v3/clusters/VsD8CdP8THeB2QIueeI38A/topics/_confluent-command/partitions"
21      }
22    }
23  ]
24 }
```

4. Describe topic

The screenshot shows the Postman interface with a GET request to `http://localhost:8082/v3/clusters/VsD8CdP8THeB2QIueeI38A/topics/victor_topic`. The request is successful, returning a 200 OK status. The response body is a JSON object describing the topic `"victor_topic"`.

```
1 {
2   "kind": "KafkaTopic",
3   "metadata": {
4     "self": "http://rest-proxy:8082/v3/clusters/VsD8CdP8THeB2QIueeI38A/topics/victor_topic",
5     "resource_name": "crn:///kafka-VsD8CdP8THeB2QIueeI38A/topic=victor_topic"
6   },
7   "cluster_id": "VsD8CdP8THeB2QIueeI38A",
8   "topic_name": "victor_topic",
9   "is_internal": false,
10  "replication_factor": 1,
11  "partitions_count": 1,
12  "partitions": {
13    "related": "http://rest-proxy:8082/v3/clusters/VsD8CdP8THeB2QIueeI38A/topics/victor_topic/partitions"
14  },
15  "config": {
16    "related": "http://rest-proxy:8082/v3/clusters/VsD8CdP8THeB2QIueeI38A/topics/victor_topic/configs"
17  },
18  "partition_reassignments": {
19    "related": "http://rest-proxy:8082/v3/clusters/VsD8CdP8THeB2QIueeI38A/topics/victor_topic/partitions/-/reassignment"
20  }
21 }
```

5. Create consumer group

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 "My first collection" with two folders: "First folder inside collection" and "Second folder inside collection". The main panel displays a POST request to `http://localhost:8082/consumers/MyGroup`. The request is configured with the following headers:

Header	Value
Accept	*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive
Content-Type	application/vnd.kafka.v2+json

The request body is in JSON format, showing the following structure:

```
1 {
2   "instance_id": "MyConsumer",
3   "base_uri": "http://rest-proxy:8082/consumers/MyGroup/instances/MyConsumer"
4 }
```

The status bar at the bottom indicates a successful response with status 200 OK, time 23 ms, and size 291 B.

6. Create subscription

The screenshot shows the Postman interface with the same workspace "My Workspace". The "Collections" tab is active, showing the same collection structure. The main panel displays a POST request to `http://localhost:8082/consumers/MyGroup/instances/MyConsumer/subscription`. The request is configured with the following headers:

Header	Value
Accept	*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive
Content-Type	application/vnd.kafka.v2+json

The request body is in JSON format, showing the following structure:

```
1 {
```

The status bar at the bottom indicates a response with status 204 No Content, time 18 ms, and size 64 B.

7. Send data

The screenshot shows the Postman interface with a POST request configured to send data to a Kafka topic. The URL is `http://localhost:8082/v3/clusters/VsD8CdP8TheB2Qiuue1J8A/topics/victor_topic/records`. The request headers are set to `Accept`, `Accept-Encoding`, `Connection`, and `Content-Type` (application/json). The body is a JSON object representing a Kafka record.

```
POST http://localhost:8082/v3/clusters/VsD8CdP8TheB2Qiuue1J8A/topics/victor_topic/records
```

Headers (9):

- Accept
- Accept-Encoding: gzip, deflate, br
- Connection: keep-alive
- Content-Type: application/json

Body (JSON):

```
{  "cluster_id": "VsD8CdP8TheB2Qiuue1J8A",  "topic_name": "victor_topic",  "partition_id": 0,  "offset": 1,  "timestamp": "2023-11-06T21:17:32.621Z",  "key": {    "type": "JSON",    "size": 6  },  "value": {    "type": "JSON",    "size": 13  }}
```

Status: 200 OK Time: 17 ms Size: 376 B Save Response

8. Get messages

The screenshot shows the Postman interface with a GET request configured to retrieve messages from a Kafka topic. The URL is `http://localhost:8082/consumers/MyGroup/instances/MyConsumer/records`. The request headers are set to `Content-Type` (application/vnd.kafka.v2-json) and `Accept` (application/vnd.kafka.json.v2+json). The body is a JSON array of Kafka records.

```
GET http://localhost:8082/consumers/MyGroup/instances/MyConsumer/records
```

Headers (8):

- Content-Type: application/vnd.kafka.v2-json
- Accept: application/vnd.kafka.json.v2+json

Body (JSON):

```
[  {    "topic": "victor_topic",    "key": "key1",    "value": "value1 Buna",    "partition": 0,    "offset": 0  },  {    "topic": "victor_topic",    "key": "key1",    "value": "value1 Buna",    "partition": 0,    "offset": 1  }]
```

Status: 200 OK Time: 1016 ms Size: 425 B Save Response