In memory key value datastore using GOLANG

A key-value database (sometimes called a key-value store) uses a simple key-value method to store data. These databases contain a simple string (the key) that is always unique and an arbitrary large data field (the value). They are easy to design and implement.

Phone directory

Key	Value
Paul	(091) 9786453778
Greg	(091) 9686154559
Marco	(091) 9868564334

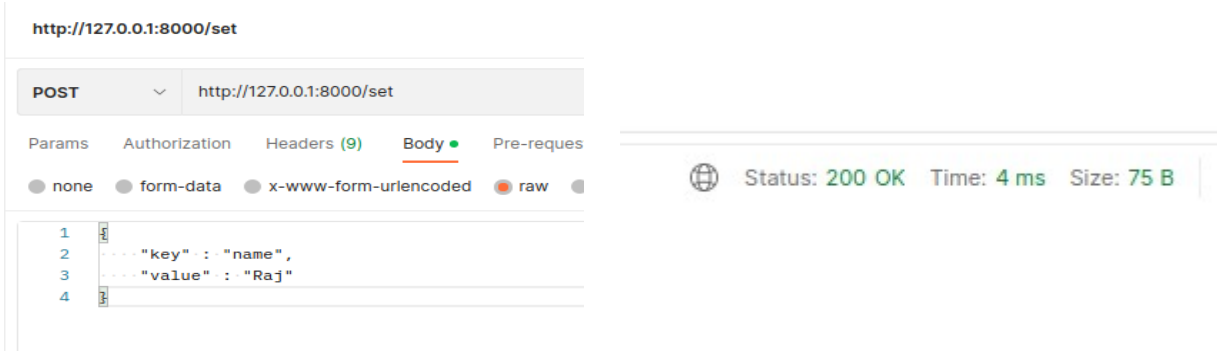
MAC table

Key	Value
10.94.214.172	3c:22:fb:86:c1:b1
10.94.214.173	00:0a:95:9d:68:16
10.94.214.174	3c:1b:fb:45:c4:b1

- I have created a simple in memory data store that performs operations on it and uses REST API for communication.

```
Supported Endpoints are:
POST                /set
GET                 /get/:key
POST                /qpush
GET                 /qpop/:list
```

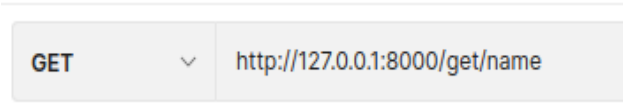
/set - This endpoint is a POST request that stores key value pair in the defined map data structure and gives appropriate http status code for response.

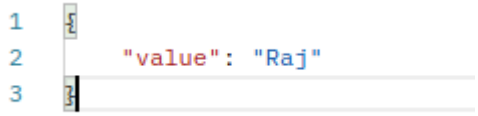


CURL CODE-SNIPPET

```
curl --location --request POST 'http://127.0.0.1:8000/set' \
--header 'Content-Type: text/plain' \
--data-raw '{
    "key" : "name",
    "value" : "Raj"
}'
```

/get - This endpoint is a GET request that retrieves the value of a key saved in the in memory datastore.

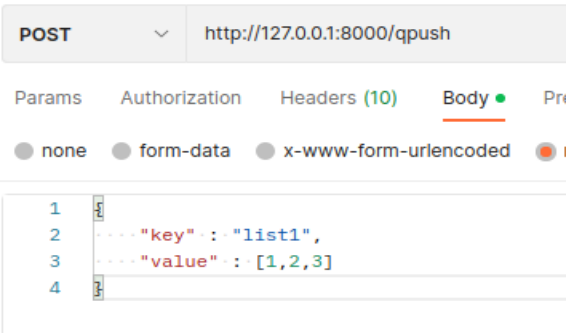




CURL CODE-SNIPPET

```
curl --location --request GET 'http://127.0.0.1:8000/get/name'
```

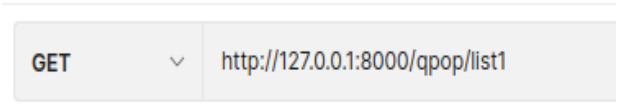
/qpush – This endpoint is a POST request that saves key-value data in a queue like structure.



CURL CODE-SNIPPET

```
curl --location --request POST 'http://127.0.0.1:8000/qpush' \
--header 'Content-Type: application/json' \
--data-raw '{
    "key" : "list1",
    "value" : [1,2,3]
}'
```

/qpop - This endpoint is a GET request that retrieves stored list data associated with a key.




```
{
    "value": 3
}
```

CURL CODE-SNIPPET

```
curl --location --request GET 'http://127.0.0.1:8000/qpop/list1'
```

Validation Checks are implented to handle cases like:

- Adding value without key/value
- Accessing value of a non-existent key
- Accessing non-existent queue
- Popping elements from a queue which is empty
- Using Different format of data

```
{
    "error": "queue is empty"
}
```

```
{
    "error": "queue not found"
}
```

```
{
    "error": "Key: 'setFormat.Key' Error:Field validation for 'Key' failed on the 'required' tag"
}
```

```
{
    "error": "Key: 'setFormat.Value' Error:Field validation for 'Value' failed on the 'required' tag"
}
```

```
{
    "error": "key not found"
}
```