CumuLogic NoSQL API Guide

June 2014

Table of Contents

1. Entity: NoSQL Instance

1.1 Create NoSQL Instance

Http Method	URL	Description
POST	/NoSQL/api/v1/NoSQLinstance	Creates a new NoSQL instance

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name "**create**"	Yes
Name	String	Unique instance name	Yes
Description	String	Description about VM	No

engineID	Integer	Unique ID of existing engine
		Ex. "engine":
		{
		"id": "1"
		}
			Yes

Nodes	Integer	Number of instances want to provision	Yes

targetCloudID	Integer	Unique ID of created target cloud
		Ex. "targetCloud":
		{
		"id": "1"
		}
			Yes

InstanceTypeID	Integer	Unique ID of instance type
		Ex. "instanceType":
		{
		"id": "1"
		}
			Yes

availabilityZone	String	Name of zone desired to be	Yes

		launched If targetcloud is VM pool then it is not mandatory	
serviceTag	String	Environment type testing, development, production, staging	Yes

accessGroupId	Integer	Unique ID of created service access group
		Ex. "serviceAccessGroup":
		{
		"id": "1"
		}
			No

collectionName	String	Collection name	Yes
databaseName	String	Database name	Yes
masterUserName	Integer	Database user name	Yes
masterUserPassword	Integer	Database password	Yes
characterSet	String	Character set format UTF-8	No
port	Integer	Database default port 3036	Yes

parameterGroupID	Integer	Unique ID of created service access group
		Ex. "parameterGroup":
		{
		"id": "1"
		}
			No

storageSize	Integer	The size of the volume Valid values: 10-1024	Yes
isShardingEnable	Integer	If want to enable sharding Valid Values: 0 If enabling of sharding is not desired 1 if want to enable sharding	Yes

		Deafult value 0	
isShardKeyHashEnable	Integer	If want to enable shard key hash Valid Values: 0 – disable it 1 - enable it Default value 0	No
isAutomaticHashOnIdEnable	Integer	If want to enable automatic hash on ID Valid Values: 0 – disable it 1 - enable it Default value 0	Yes if IsShardingEnable =1
shardKey	String	Shard key value	Yes if IsShardingEnable =1 and isAutomaticHashOnIdEnable =1
shardingArchitecture	String	Type of sharding architecture 1] Testing 2]Production	Yes if IsShardingEnable =1
isbackupPreferenceSet	Integer	If want to set backup Valid Values: 0 If backup is not desired 1 if want to set backup Default value 0	No
backupRetentionP	Integer	After how many days wants to take	No

eriod		backup Values in days Default: 1 Day	
backupStartTime	String	When to start automated backup time Format: HH:MM	No

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlinstance" -X POST -H "CL-AUTH-TOKEN:b4ce975b-d326-44d3-b3bc-b739fd8c9bea" -d'{"create": {"noSqlInstance": {"name": "test123","nodes": "1","masterUserName": "demo","masterUserPassword": "demodemo","engine":{"id": "1"},"targetCloud":{"id": "4"},"availabilityZone": "az-1.region-a.geo-1","instanceType":{"id": "1"},"serviceTag": "Testing","databaseName":"test","collectionName":"test1","storageSize": "10","port":"27017" ,"serviceAccessGroup":{"id": "4010"},"shardingArchitecture": "Testing","shardKey": "Testing","isShardingEnable": "0","isAutomaticHashOnIdEnable": "0","isShardKeyHashEnable": "0"},"isDefault": false,"isProvision": true}}'

Example Response:

{"success":true,"errors":{},"response":{"noSqlInstances":[{"accessGroupName":null,"port":27017,"autoMinorEnabled":2,"status":2,"backupRetentionPeriod":null,"backupStartTime":null,"charset":null,"engineName":"MongoDB-2.2.0","targetCloudName":"targe1234sfsd4567","instanceType":"xsmall","id":7,"serviceTag": "Testing","createdBy":43,"description":null,"nodes":1,"name":"test123","masterPassword":"demodemo","storageSize":10,"isInprogress":2,"createdDate":"2014-04-16 02:27:05","masterUserName":"demo"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)

1.2 Update NoSQL Instance

Http Method	URL	Description
PUT	/NoSQL/api/v1/ NoSQLinstance	Updates the existing SQL instance

Request Parameters:

Attribute	Type	Description	Required
Id	Integer	Unique NoSQL instance ID	
Name	String	Unique instance name	Yes
Description	String	Description about VM	No

engineId	Integer	Unique ID of existing engine
		
		Ex. "engine":
		
		{
		"id": "1"
		}
			Yes

Nodes	Integer	Number of instances want to provision	Yes

targetCloudId	Integer	Unique ID of created target cloud
		Ex. "targetCloud":
		{
		"id": "1"
		}
			Yes

instanceTypeId	Integer	Unique ID of instance type
		Ex. "instanceType":
		{
		"id": "1"
		}
			Yes

availabilityZone	String	Name of zone desired to be launched If targetcloud is VM pool then it not mandatory	Yes
serviceTag	String	Environment type testing, development, production, staging	Yes

accessGroupId	Integer	Unique ID of created service access group
		Ex. "serviceAccessGroup":
		{
		"id": "1"
		}
			No

collectionName	String	Collection name	No
databaseName	String	Database name	No
masterUserName	Integer	Database user name	Yes
masterUserPassword	Integer	Database password	Yes
characterSet	String	Character set format UTF-8	No
Port	Integer	Database default port 3036	No
parameterGroupId	Integer	Unique ID of created service access group **Ex.** "parameterGroup": {	No

		"id": "1" }	
storageSize	Integer	The size of the volume Valid values: 10-1024	Yes
isShardingEnable	Integer	If want to enable sharding Valid Values: 0 if sharding is undesired 1 if want to enable sharding Default value 0	Yes
isShardKeyHashEnable	Integer	If want to enable shard key hash Valid Values: 0 – disable it 1 - enable it Default value 0	No
isAutomaticHashOnIdEnable	Integer	If want to enable automatic hash on ID Valid Values: 0 – disable it 1 - enable it Default value 0	Yes if IsShardingEnable=1
shardKey	String	Shard key value	Yes if IsShardingEnable=1 and isAutomaticHashOnIdEnable =1
shardingArchitecture	String	Type of Sharding Architecture 1] Testing 2]Production	Yes if IsShardingEnable=1

isbackupPreferenceSet	Integer	If want to set backup. Valid Values: 0 If backup is undesired 1 if want to set backup Default value 0	No
backupRetentionPeriod	Integer	After how many days wants to take backup Values in days Default: 1 Day	No
backupStartTime	String	When to start automated backup time Format: HH:MM	No

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlinstance/1" -X PUT -H "CL-AUTH-TOKEN:7f4b8754-3442-4569-bccf-ab3a40451106" -d'{"name": "updatetest","nodes": "3","masterUserName": "demo","masterUserPassword": "demodemo","engine":{"id": "1"},"targetCloud":{"id": "4"},"availabilityZone":"az-1.region-a.geo-1","instanceType":{"id": "1"},"serviceTag": "Testing","storageSize": "10" ,"serviceAccessGroup":{"id": "1"},"shardingArchitecture": "Testing","shardKey": "Testing","isShardingEnable": "0","isAutomaticHashOnIdEnable": "0","isShardKeyHashEnable": "0"}'

Example Response: JSON

{"success":true,"errors":{},"response":{}}

Response Code(s):

Normal Response Code(s):	200

Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), invalidRequest(1000)

1.3 Modify NoSQL Instance

Http Method	URL	Description
POST	http:/NoSQL/api/v1/ NoSQL instance	Modify the existing NoSQL instance

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name "**modify**"	Yes
Id	Integer	Unique ID of the existing instance	Yes

parameterGroupId	Integer	Unique ID of created service access group
		Ex. "parameterGroup":
		{
		"id": "1"
		}
			Yes

isbackupPreferenceSet	Integer	If want to set backup Valid Values: 0 If backup is undesired 1 if want to set backup	No

backupRetentionPeriod	Integer	After how many days wants to take backup Values in days Default: 1 Day	No
backupStartTime	String	When to start automated backup time Ex: HH:MM	No
applyNow	Boolean	Apply modified changes to instance Values: true – If wants to apply changes on instance quickly false – if don't want to apply changes	Yes

Example Request:

curl -i "http://localhost:8080/nosql/api/v1/nosqlinstance" -X POST -H "CL-AUTH-TOKEN:fdfdb51b-e10b-4f28-a018-4036719750ed" -d'{"modify":{"noSqlInstance":{"id": "3","isbackupPreferenceSet": "1","backupRetentionPeriod": "1","backupStartTime": "01:00 ","parameterGroup": {"id": "1"}},"applyNow": "true"}}'

Example Response:

{"success":true,"errors":{},"response":{"noSqlInstances":[{"multiaz":null,"port":27017,"autoMinorEnabled":2,"backupRetentionPeriod":1,"backupStartTime":"01:00 ","isbackupPreferenceSet":1,"isInProgress":2,"noOfSnapshots":null,"instancetype":{"id":1,"instanceTypeId":"179","name":"Xsmall"},"id":3,"serviceTag":"Testing","description":"abcv","masterPassword":"demodemo","name":"abcd","availabilityZone":"1","isArbiterAdded":null,"createdDate":"2014-05-18 13:43:13","masterUserName":"demo","parametergroup":{"id":1,"parameterGroupFamilyName":"MongoDB","name":"paramgrp","parameterGroupFamilyId":null},"collectionName":"demo","serviceaccessgroup":{"id":4300,"name":"default_MongoDB_1"},"targetcloud":{"id":1,"name":"CONTEGIX_TC"},"engine":{"id":1,"engineName":"MongoDB","version":"2.2.0"},"status":1,"charset":"UTF-8","databaseName":"demo","environment":"Testing","createdBy":2,"nodes":1,"storageSize":10}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), invalidRequest(1000)

1.4 Delete NOSQL Instance

Http Method	URL	Description
DELETE	/NoSQL/api/v1/NoSQLinstance /<id>	Deletes the existing NoSQL instance

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlinstance/30" -X DELETE -H "CL-AUTH-TOKEN:7f4b8754-3442-4569-bccf-ab3a40451106"

Response Format:

{"success":true,"errors":{},"response":{}}

Response Code(s):

Normal Response Code(s):	204
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401),

	forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidParameterValue(1013), invalidRequest(1000)

1.5 Get All NoSQL Instances

Http Method	URL	Description
GET	/NoSQL/api/v1/nNoSQLinstance	To get all NoSQL instances

Example Request:

curl -i "http://localhost:8080/nosql/api/v1/nosqlinstance" -X GET -H "CL-AUTH-TOKEN:fee1f160-d184-4381-9425-10b653cfce32"

Example Response:

{"success":true,"errors":{},"response":{"noSqlInstances":[{"multiaz":null,"port":27017,"autoMinorEnabled":2,"backupRetentionPeriod":null,"backupStartTime":null,"isbackupPreferenceSet":null,"isInProgress":2,"noOfSnapshots":0,"instancetype":{"id":1,"instanceTypeId":"179","name":"Xsmall"},"id":1,"serviceTag":2,"description":"NoSQLinstance","masterPassword":"demodemo","name":"NoSQLinstance","availabilityZone":"1","createdDate":"2014-05-18 10:41:27","masterUserName":"demo","collectionName":"demo","serviceaccessgroup":{"id":4297,"name":"default_MongoDB_644"},"targetcloud":{"id":1,"name":"CONTEGIX_TC"},"engine":{"id":1,"engineName":"MongoDB","version":"2.2.0"},"status":2,"charset":"UTF-8","databaseName":"demo","environment":"Testing","createdBy":2,"nodes":1,"storageSize":10}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503),

	unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

1.6 Get NoSQL Instance by ID

Http Method	URL	Description
GET	/NoSQL/api/v1/NoSQLinstance/<id>	To get NoSQL instance by ID

Query Parameters:

Filter Name	Type	Description	Required
ID	Integer	Unique ID of the instance	Yes

Example Request:

curl -i "http://localhost:8080/nosql/api/v1/nosqlinstance/1" -X GET -H "CL-AUTH-TOKEN:fee1f160-d184-4381-9425-10b653cfce32"

Example Response:

{"success":true,"errors":{},"response":{"noSqlInstances":[{"multiaz":null,"port":27017,"autoMinorEnabled":2,"backupRetentionPeriod":null,"backupStartTime":null,"isbackupPreferenceSet":null,"isInProgress":2,"noOfSnapshots":0,"instancetype":{"id":1,"instanceTypeId":"179","name":"Xsmall"},"id":1,"serviceTag":2,"description":"NoSQLinstance","masterPassword":"demodemo","name":"NoSQLinstance","availabilityZone":"1","createdDate":"2014-05-18 10:41:27","masterUserName":"demo","collectionName":"demo","serviceaccessgroup":{"id":4297,"name":"default_MongoDB_644"},"targetcloud":{"id":1,"name":"CONTEGIX_TC"},"engine":{"id":1,"engineName":"MongoDB","version":"2.2.0"},"status":2,"charset":"UTF-8","databaseName":"demo","environment":"Testing","createdBy":2,"nodes":1,"storageSize":10}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

1.7 Get default NoSQL Instance

Http Method	URL	Description
GET	/ NoSQL /api/v1/ NoSQL instance?userId=<userId>&default=<1 or 0>	To get NoSQL instance by ID

Query Parameters:

Filter Name	Type	Description	Required
userID	Integer	Unique ID of the user	Yes
Default	Integer	Default value is 1	Yes

Example Request:

$ curl -i "http:// localhost:8080/nosql/api/v1/nosqlinstance?userId=43&default=1" -X GET -H "CL-AUTH-TOKEN:2d5f0fd5-e144-4c7d-b155-5e61c3ea387f"

Example Response:

{"success":true,"errors":{},"response":{"noSqlInstances":[{"accessGroupName":null,"port":27017,"autoMinorEnabled":2,"status":2,"backupRetentionPeriod":null,"backupStartTime":null,"charset":null,"engineName":"MongoDB-2.2.0","targetCloudName":"falcon_hpcs","instanceType":"xsmall","id":31,"serviceTag":"Testing","createdBy":43,"description":"default_instance","nodes":1,"name":"default_instance","ma

sterPassword":"demodemo","storageSize":10,"isInprogress":2,"createdDate":"2014-04-25 06:31:30","masterUserName":"demo"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

1.8 Provision NoSQL Instance

Http Method	URL	Description
POST	/NoSQL/api/v1/ NoSQLinstance	Starts provisioning of the existing NoSQL instance.

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name "**provision**"	Yes
ID	Integer	Unique ID of the existing instance	Yes

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlinstance" -X POST -H "CL-AUTH-TOKEN:7f4b8754-3442-4569-bccf-ab3a40451106" -d'{"provision": { "noSqlInstance": { "id": "1"}}}'

Example Response:

{"success":true,"errors":{},"response":{}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), invalidRequest(1000)

1.9 Terminate NoSQL Instance

Http Method	URL	Description
POST	/NoSQL/api/v1/NoSQLinstance	Terminate the existing NoSQL instance

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name "**terminate**"	Yes
ID	Integer	Unique ID of the existing instance	Yes

Example Request:

$ curl -i "http:// localhost:8080/nosql/api/v1/nosqlinstance" -X POST -H "CL-AUTH-TOKEN:af169236-ed07-41a9-bbaf-47f779df01c3" -d'{"terminate": {"noSqlInstance": {"id": "1"}}}'

Example Response: JSON

{"success":true,"errors":{},"response":{}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), invalidRequest(1000)

1.10 Restart NoSQL Instance

Http Method	URL	Description
POST	/NoSQL/api/v1/NoSQLinstance	Restart the existing NoSQL instance

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name "**restart**"	Yes
ID	Integer	Unique ID of the existing instance	Yes

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlinstance" -X POST -H "CL-AUTH-TOKEN:fee1f160-d184-4381-9425-10b653cfce32" -d'{"restart": {"noSqlInstance": {"id": "2"}}}'

Example Response:

{"success":true,"errors":{},"response":{}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), invalidRequest(1000)

1.11 Clone NoSQL Instance

Http Method	URL	Description
POST	/NoSQL/api/v1/NoSQLinstance	Cancel provision of the existing NoSQL instance

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name "**clone**"	Yes
ID	Integer	Unique ID of the existing instance	Yes

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlinstance" -X POST -H "CL-AUTH-TOKEN:372c5865-1866-4efa-b6d8-ce5963a2850d" -d'{"clone": {"noSqlInstance": {"id": "1"}}}'

Example Response:

{"success":true,"errors":{},"response":{"noSqlInstances":[{"multiaz":null,"port":27017,"autoMinorEnabled":2,"backupRetentionPeriod":null,"backupStartTime":null,"isbackupPreferenceSet":null,"isInProgress":2,"noOfSnapshots":0,"instancetype":{"id":1,"instanceTypeId":"179","name":"Xsmall"},"id":8,"serviceTag":"Testing","description":"NoSQLinstance","masterPassword":"demodemo","name":"cloned-NoSQLinstance-87","availabilityZone":"1","createdDate":"2014-05-18 10:41:27","masterUserName":"demo","collectionName":"demo","serviceaccessgroup":{"id":4297,"name":"default_MongoDB_644"},"targetcloud":{"id":1,"name":"CONTEGIX_TC"},"engine":{"id":1,"engineName":"MongoDB","version":"2.2.0"},"status":2,"charset":"UTF-8","databaseName":"demo","environment":"Testing","createdBy":2,"nodes":1,"storageSize":10}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), invalidRequest(1000)

1.12 Add Node

Http Method	URL	Description
POST	/NoSQL/api/v1/NoSQLinstance	Add the new node to existing NoSQL instance

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name “**addNode**”	Yes
ID	Integer	Unique ID of the existing instance	Yes

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlinstance" -X POST -H "CL-AUTH-TOKEN:372c5865-1866-4efa-b6d8-ce5963a2850d" -d'{"addNode": {"noSqlInstance": {"id": "1"}}}'

Example Response:

{"success": true,"errors": {},"response": {" noSqlInstances": [{"accessGroupName":"","port":null,"autoMinorEnabled":2,"status":2,"backupRetentionPeriod":null,"backupStartTime":null,"charset":null,"engineName":"MongoDB-2.2.0","targetCloudName":"targetcloud1","instanceType":null,"id":15,"environment":"Testing","createdBy":2,"nodes":1,"description":"Testing","masterPassword":"demodemo","name":"cloned_Test","storageSize":10,"isInprogress":2,"masterUserName":"demo","createdDate":"2014-03-26 14:26:04"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), invalidRequest(1000)

1.13 Remove Node

Http Method	URL	Description
POST	/NoSQL/api/v1/NoSQLinstance	Remove the added node from the existing NoSQL instance

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name "**removeNode**"	Yes
ID	Integer	Unique ID of the existing instance	Yes

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlinstance" -X POST -H "CL-AUTH-TOKEN:372c5865-1866-4efa-b6d8-ce5963a2850d" -d'{"removeNode": {"noSqlInstance": {"id": "1"}}}'

Example Response:

{"success": true,"errors": {},"response": {" noSqlInstances": [{"accessGroupName":"","port":null,"autoMinorEnabled":2,"status":2,"backupRetentionPeriod":null,"backupStartTime":null,"charset":null,"engineName":"MongoDB-2.2.0","targetCloudName":"targetcloud1","instanceType":null,"id":15,"environment":"Testing","createdBy":2,"nodes":1,"description":"Testing","masterPassword":"demodemo","name":"cloned_Test","storageSize":10,"isInprogress":2,"masterUserName":"demo","createdDate":"2014-03-26 14:26:04"}]}}

Response Code(s):

Normal Response Code(s):	200

Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), invalidRequest(1000)

2. Entity: NoSQL Snapshot

2.1 Snapshot

Http Method	URL	Description
POST	/NoSQL/api/v1/NoSQLsnapshot	Creates a new snapshot of the instance

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name "**create**"	Yes
name	String	Unique instance name	Yes
description	String	Description about VM	No
backupStartTime	String	When to start automated backup time Format: HH:MM	No

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlsnapshot" -X POST -H "CL-AUTH-TOKEN:39c9e0f3-e6cb-4ab7-a98a-b026338e2fc1" -d'{"create": {"noSqlSnapshot":{"noSqlInstance": {"id": "5"},"name": "demo_snapshot"}}}'

Example Response:

{"success":true,"errors":{},"response":{"noSqlSnapshots":[{"id":1,"name":"demo_snapshot","type":"manual"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)

2.2 RestoreFromSnapshot

Http Method	URL	Description
POST	/NoSQL/api/v1/NoSQLsnapshot	Restore the instance from snapshot

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name "**restoreFromSnapshot**"	Yes
instanceName	String	Unique instance name	Yes
ID	Integer	Unique ID of the snapshot	Yes
availabilityZone	String	When to start automated backup time Format: HH:MM	Yes
availabilityZoneName	String	Availability zone name	Yes
Port	Integer	Port of database	Yes

instanceType	Array	Instance type ID
		Ex.,
		"instanceType": {
		"id": "1"
		}
			Yes

Example Request:

curl -i "http://localhost:8080/nosql/api/v1/nosqlsnapshot" -X POST -H "CL-AUTH-TOKEN:bb63a197-a4f5-4218-887a-0a08afdff473" -d'{"restoreFromSnapshot": {"noSqlSnapshot": {"id": "1", "instanceName": "RestoreFromSnapy","availabilityZone": "ContZone","availabilityZoneName": "cont","port": "27017", "instanceType": {"id": "31"},"storageSize": "10","isAutoUpdateEnabled": "1"}}}'

Example Response:

{"success":true,"errors":{},"response":{"noSqlSnapshots":[{"name":"demo_snapshot","type": "manual","id":6}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)

2.3 Get NoSQL Snapshot By Name(s)

Http Method	URL	Description
GET	/NoSQL/api/v1/NoSQLsnapshot?name=<name>	To get specific snapshot by name

Example Request:-

curl -i "http:// localhost:8080/nosql/api/v1/nosqlsnapshot?name=TestSnapshotd13" -X GET -H "CL-AUTH-TOKEN:3d765647-03b3-4334-9a3e-b2d562b5909e"

Example Response:

{"response":{"noSqlSnapshots":[{"id":9,"name":"TestSnapshotd13","noSqlInstanceId":18,"noSqlInstanceName":"tecst","type":"manual"}]},"errors":{},"success":true}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

2.4 Get NoSQL Snapshot By ID

Http Method	URL	Description
GET	/NoSQL/api/v1/NoSQLsnapshot/<id>	To get specific snapshot by ID

Example Request:

curl -i "http:// localhost:8080/nosql/api/v1/nosqlsnapshot/9" -X GET -H "CL-AUTH-TOKEN:3d765647-03b3-4334-9a3e-b2d562b5909e"

Example Response:
{"response":{"noSqlSnapshots":[{"id":9,"name":"TestSnapshotd13","noSqlInstanceId":18,"noSqlInstanceName":"tecst","type":"manual"}]},"errors":{},"success":true}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)

Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

2.5 Get NoSQL Snapshot By User

Http Method	URL	Description
GET	/NoSQL/api/v1/NoSQLsnapshot?userId=<userid>	To get specific snapshot by user ID

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlsnapshot?userId=2" -X GET -H "CL-AUTH-TOKEN:fee1f160-d184-4381-9425-10b653cfce32"

Example Response:

{"response":{"noSqlSnapshots":[{"id":9,"name":"TestSnapshotd13","noSqlInstanceId":18,"noSqlInstanceName":"tecst","type":"manual"}]},"errors":{},"success":true}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

2.6 Delete NoSQL Snapshot

Http Method	URL	Description
DELETE	/NoSQL/api/v1/NoSQLsnapshot/<id>	Deletes the existing NoSQL snapshot

Example Request:

curl -i "http://localhost:8080/nosql/api/v1/nosqlsnapshot/2" -X DELETE -H "CL-AUTH-TOKEN:3d765647-03b3-4334-9a3e-b2d562b5909e"

Response Format:

{"response": [],"errors": {},"success": true}

Response Code(s):

Normal Response Code(s):	204
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidParameterValue(1013), invalidRequest(1000)

3. Entity: NoSQL Engine

3.1 Get All NoSQL Engines

Http Method	URL	Description
GET	/NoSQL/api/v1/NoSQLengine	To get all NoSQL engines

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlengine" -X GET -H "CL-AUTH-TOKEN:fee1f160-d184-4381-9425-10b653cfce32"

Example Response:

{"success":true,"errors":{},"response":{"engines":[{"id":1,"hostType":7,"location":"/data/mongodb","name":"MongoDB","type":1,"version":"2.2.0"},{"id":2,"hostType":12,"location":"/opt/couchbase","name":"Couchbase","type":2,"version":"2.2.0"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

3.2 Get NoSQL Engine By ID

Http Method	URL	Description
GET	/NoSQL/api/v1/NoSQLengine/<id>	To get NoSQL engine by ID

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlengine/1" -X GET -H "CL-AUTH-TOKEN:fee1f160-d184-4381-9425-10b653cfce32"

Example Response:

{"success":true,"errors":{},"response":{"engines":[{"id":1,"hostType":7,"location":"/data/mongodb","name":"MongoDB","type":1,"version":"2.2.0"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

3.3 Get NoSQL Engine By Name and Version

Http Method	URL	Description
GET	/NoSQL/api/v1/NoSQLengine?name=<name>&version=<version>	To get NoSQL engine by ID

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlengine?name=MongoDB&version=2.2.0" -X GET -H "CL-AUTH-TOKEN:fee1f160-d184-4381-9425-10b653cfce32"

Example Response:

{"success":true,"errors":{},"response":{"engines":[{"id":1,"hostType":7,"location":"/data/mongodb","name":"MongoDB","type":1,"version":"2.2.0"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

4. Entity: NoSQL Parameter Groups

4.1 Create NoSQL Parameter Group

Http Method	URL	Description
POST	/NoSQL/api/v1/NoSQLparametergroup	Creates a new NoSQL Parameter Group

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name "**create**"	Yes
Name	String	Unique instance name	Yes
Description	String	Description about VM	No
parameterGroupFamilyID	Integer	Unique ID of existing parameter group family	Yes

Example Request:

curl -i "http://localhost:8080/nosql/api/v1/nosqlparametergroup" -X POST -H "CL-AUTH-TOKEN:bb63a197-a4f5-4218-887a-0a08afdff473" -d'{"create": {"description":"sacas", "name":"sacas", "parameterGroupFamilyId":"1"}}'

Example Response:

{"success":true,"errors":{},"response":{"parameterGroups":[{"id":6,"description":"sacas","name":"sacas","engineName":"MongoDB-2.2.0","parameters":[{"id":37,"scope":"Global","possibleValues":"false","name":"fork","value":null,"isDynamic":0,"comment":"Enables a daemon mode for mongod and allows you to run the database as a conventional server","defaultValue":"true","type":"boolean"},{"id":38,"scope":"Global","possibleValues":"false","name":"quiet","value":null,"isDynamic":0,"comment":"This disables all but the most critical entries in output/log file","defaultValue":"true","type":"boolean"},{"id":39,"scope":"Global","possibleValues":"false","name":"journal","value":null,"isDynamic":0,"comment":"Ensures single instance write-durability","defaultValue":"true","type":"boolean"},{"id":40,"scope":"Global","possibleValues":"false","name":"nounixsocket","value":null,"isDynamic":0,"comment":"Disables the UNIX Socket","defaultValue":"true","type":"boolean"},{"id":41,"scope":"Global","possibleValues":"false","name":"auth","value":null,"isDynamic":0,"comment":"Enables the authentication system within MongoDB","defaultValue":"true","type":"boolean"},{"id":42,"scope":"Global","possibleValues":"false","name":"verbose","value":null,"isDynamic":0,"comment":"Enables a verbose logging mode that modifies mongod output and increases logging to include a greate","defaultValue":"true","type":"boolean"},{"id":43,"scope":"Global","possibleValues":"0,1,2,3,7","name":"diaglog","value":null,"isDynamic":0,"comment":"Enables diagnostic logging. Level 3 logs all read and write options.","defaultValue":"3","type":"Numeric"},{"id":44,"scope":"Global","possibleValues":"false","name":"objcheck","value":null,"isDynamic":0,"comment":"Forces mongod to validate all requests from clients upon receipt.","defaultValue":"true","type":"boolean"},{"id":45,"scope":"Global","possibleValues":"false","name":"cpu","value":null,"isDynamic":0,"comment":"Forces mongod to report the percentage of the last interval spent in write-lock.","defaultValue":"true","type":"boolean"}],"createdDate":"2014-05-21 10:22:54"}]}}

Response Code(s):

Normal Response Code(s):	200

Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)

4.2 Delete NoSQL Parameter Group

Http Method	URI	Description
DELETE	/NoSQL/api/v1/ NoSQLparametergroup /<id>	Deletes the existing NoSQL parameter group

Example Request:

curl -i "http:// localhost:8080/nosql/api/v1/nosqlparametergroup/6" -X DELETE -H "CL-AUTH-TOKEN:3d765647-03b3-4334-9a3e-b2d562b5909e"

Response Format:

{"response": [],"errors": {},"success": true}

Response Code(s):

Normal Response Code(s):	204
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400),

	badMethod (405), overLimit (413), itemNotFound (404)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidParameterValue(1013), invalidRequest(1000)

4.3 Get NoSQL Parameter Group by ID

Http Method	URL	Description
GET	/NoSQL/api/v1/NoSQLparametergroup/<id>	To get NoSQL parameter group instance by ID

Query Parameters:

Filter Name	Type	Description	Required
ID	Integer	Unique ID of the instance	Yes

Example Request:

$ curl -i "http:// localhost:8080/nosql/api/v1/nosqlparametergroup/6" -X GET -H "CL-AUTH-TOKEN:372c5865-1866-4efa-b6d8-ce5963a2850d"

Example Response:

{"success":true,"errors":{},"response":{"parameterGroups":[{"id":6,"description":"sacas","name":"sacas","engineName":"MongoDB-2.2.0","parameters":[{"id":37,"scope":"Global","possibleValues":"false","name":"fork","value":null,"isDynamic":0,"comment":"Enables a daemon mode for mongod and allows you to run the database as a conventional server","defaultValue":"true","type":"boolean"},{"id":38,"scope":"Global","possibleValues":"false","name":"quiet","value":null,"isDynamic":0,"comment":"This disables all but the most critical entries in output/log file","defaultValue":"true","type":"boolean"},{"id":39,"scope":"Global","possibleValues":"false","name":"journal","value":null,"isDynamic":0,"comment":"Ensures single instance write-

durability","defaultValue":"true","type":"boolean"},{"id":40,"scope":"Global","possibleValues":"false","name":"nounixsocket","value":null,"isDynamic":0,"comment":"Disables the UNIX Socket","defaultValue":"true","type":"boolean"},{"id":41,"scope":"Global","possibleValues":"false","name":"auth","value":null,"isDynamic":0,"comment":"Enables the authentication system within MongoDB","defaultValue":"true","type":"boolean"},{"id":42,"scope":"Global","possibleValues":"false","name":"verbose","value":null,"isDynamic":0,"comment":"Enables a verbose logging mode that modifies mongod output and increases logging to include a greate","defaultValue":"true","type":"boolean"},{"id":43,"scope":"Global","possibleValues":"0,1,2,3,7","name":"diaglog","value":null,"isDynamic":0,"comment":"Enables diagnostic logging. Level 3 logs all read and write options.","defaultValue":"3","type":"Numeric"},{"id":44,"scope":"Global","possibleValues":"false","name":"objcheck","value":null,"isDynamic":0,"comment":"Forces mongod to validate all requests from clients upon receipt.","defaultValue":"true","type":"boolean"},{"id":45,"scope":"Global","possibleValues":"false","name":"cpu","value":null,"isDynamic":0,"comment":"Forces mongod to report the percentage of the last interval spent in write-lock.","defaultValue":"true","type":"boolean"}],"createdDate":"2014-05-21 10:22:54"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

4.4 Get NoSQL Parameter Group by Name

Http Method	URL	Description
GET	/NoSQL/api/v1/ NoSQLparametergroup?name=<name>	To get NoSQL parameter group by parameter group name

Query Parameters:

Filter Name	Type	Description	Required
Name	String	Unique name of the instance	Yes

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlparametergroup?name=sacas" -X GET -H "CL-AUTH-TOKEN:372c5865-1866-4efa-b6d8-ce5963a2850d"

Example Response:

{"success":true,"errors":{},"response":{"parameterGroups":[{"id":6,"description":"sacas","name":"sacas","engineName":"MongoDB-2.2.0","parameters":[{"id":37,"scope":"Global","possibleValues":"false","name":"fork","value":null,"isDynamic":0,"comment":"Enables a daemon mode for mongod and allows you to run the database as a conventional server","defaultValue":"true","type":"boolean"},{"id":38,"scope":"Global","possibleValues":"false","name":"quiet","value":null,"isDynamic":0,"comment":"This disables all but the most critical entries in output/log file","defaultValue":"true","type":"boolean"},{"id":39,"scope":"Global","possibleValues":"false","name":"journal","value":null,"isDynamic":0,"comment":"Ensures single instance write-durability","defaultValue":"true","type":"boolean"},{"id":40,"scope":"Global","possibleValues":"false","name":"nounixsocket","value":null,"isDynamic":0,"comment":"Disables the UNIX Socket","defaultValue":"true","type":"boolean"},{"id":41,"scope":"Global","possibleValues":"false","name":"auth","value":null,"isDynamic":0,"comment":"Enables the authentication system within MongoDB","defaultValue":"true","type":"boolean"},{"id":42,"scope":"Global","possibleValues":"false","name":"verbose","value":null,"isDynamic":0,"comment":"Enables a verbose logging mode that modifies mongod output and increases logging to include a greate","defaultValue":"true","type":"boolean"},{"id":43,"scope":"Global","possibleValues":"0,1,2,3,7","name":"diaglog","value":null,"isDynamic":0,"comment":"Enables diagnostic logging. Level 3 logs all read and write options.","defaultValue":"3","type":"Numeric"},{"id":44,"scope":"Global","possibleValues":"false","name":"objcheck","value":null,"isDynamic":0,"comment":"Forces mongod to validate all requests from clients upon receipt.","defaultValue":"true","type":"boolean"},{"id":45,"scope":"Global","possibleValues":"false","name":"cpu","value":null,"isDynamic":0,"comment":"Forces mongod to report the

percentage of the last interval spent in write-lock.","defaultValue":"true","type":"boolean"}],"createdDate":"2014-05-21 10:22:54"}]}}
s

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

4.5 Get NoSQL Parameter Group by user ID

Http Method	URL	Description
GET	/NoSQL/api/v1/ NoSQLparametergroup?userId=< userID >	To get NoSQL parameter group by user ID

Query Parameters:

Filter Name	Type	Description	Required
userID	Int	User ID	Yes

Example Request:

curl -i "http:// localhost:8080/nosql/api/v1/nosqlparametergroup?userId=2"; -X GET -H "CL-AUTH-TOKEN:3d765647-03b3-4334-9a3e-b2d562b5909e"

Example Response:

{"success":true,"errors":{},"response":{"parameterGroups":[{"id":6,"description":"sacas","name":"sacas","engineName":"MongoDB-2.2.0","parameters":[{"id":37,"scope":"Global","possibleValues":"false","name":"fork","value"

:null,"isDynamic":0,"comment":"Enables a daemon mode for mongod and allows you to run the database as a conventional server","defaultValue":"true","type":"boolean"},{"id":38,"scope":"Global","possibleValues":"false","name":"quiet","value":null,"isDynamic":0,"comment":"This disables all but the most critical entries in output/log file","defaultValue":"true","type":"boolean"},{"id":39,"scope":"Global","possibleValues":"false","name":"journal","value":null,"isDynamic":0,"comment":"Ensures single instance write-durability","defaultValue":"true","type":"boolean"},{"id":40,"scope":"Global","possibleValues":"false","name":"nounixsocket","value":null,"isDynamic":0,"comment":"Disables the UNIX Socket","defaultValue":"true","type":"boolean"},{"id":41,"scope":"Global","possibleValues":"false","name":"auth","value":null,"isDynamic":0,"comment":"Enables the authentication system within MongoDB","defaultValue":"true","type":"boolean"},{"id":42,"scope":"Global","possibleValues":"false","name":"verbose","value":null,"isDynamic":0,"comment":"Enables a verbose logging mode that modifies mongod output and increases logging to include a greate","defaultValue":"true","type":"boolean"},{"id":43,"scope":"Global","possibleValues":"0,1,2,3,7","name":"diaglog","value":null,"isDynamic":0,"comment":"Enables diagnostic logging. Level 3 logs all read and write options.","defaultValue":"3","type":"Numeric"},{"id":44,"scope":"Global","possibleValues":"false","name":"objcheck","value":null,"isDynamic":0,"comment":"Forces mongod to validate all requests from clients upon receipt.","defaultValue":"true","type":"boolean"},{"id":45,"scope":"Global","possibleValues":"false","name":"cpu","value":null,"isDynamic":0,"comment":"Forces mongod to report the percentage of the last interval spent in write-lock.","defaultValue":"true","type":"boolean"}],"createdDate":"2014-05-21 10:22:54"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

4.6 Get NoSQL Parameter Group by Engine Name

Http Method	URL	Description
GET	/NoSQL/api/v1/ NoSQLparametergroup?userId=< userId >&engineName=<engineName>	To get NoSQL parameter group by user ID

Query Parameters:

Filter Name	Type	Description	Required
userID	Int	Unique User ID of the User	Yes
engineName	String	Engine name along with version	Yes

Example Request:

$ curl -i "http:// localhost:8080/nosql/api/v1/nosqlparametergroup?name=MongoDB&version=2.2.0" -X GET -H "CL-AUTH-TOKEN:372c5865-1866-4efa-b6d8-ce5963a2850d"

Example Response:

{ "success": true,"errors": {},”response": { "parameterGroups"[{ "description":"sacas", "name":"sacas", "engineName":"MongoDB 2.2.0","parameters":[{"scope":"Global","possibleValues":"false","name":"fork","value":null,"isDynamic":0,"type":"boolean","defaultValue":"true","comment":"Enables a daemon mode for mongod and allows you to run the database as a conventional server"},{"scope":"Global","possibleValues":"false","name":"quiet","value":null,"isDynamic":0,"type":"boolean","defaultValue":"true","comment":"This disables all but the most critical entries in output\/log file"},{"scope":"Global","possibleValues":"false","name":"journal","value":null,"isDynamic":0,"type":"boolean","defaultValue":"true","comment":"Ensures single instance write-durability"},{"scope":"Global","possibleValues":"false","name":"nounixsocket","value":null,"isDynamic":0,"type":"boolean","defaultValue":"true","comment":"Disables the UNIX Socket"},{"scope":"Global","possibleValues":"false","name":"auth","value":null,"isDynamic":0,"type":"boolean","defaultValue":"true","comment":"Enables the authentication system within MongoDB"},{"scope":"Global","possibleValues":"false","name":"verbose","value":null,"isDyn

amic":0,"type":"boolean","defaultValue":"true","comment":"Enables a verbose logging mode that modifies mongod output and increases logging to include a greate"},{"scope":"Global","possibleValues":"0,1,2,3,7","name":"diaglog","value":null,"isDynamic":0,"type":"Numeric","defaultValue":"3","comment":"Enables diagnostic logging. Level 3 logs all read and write
options."},{"scope":"Global","possibleValues":"false","name":"objcheck","value":null,"isDynamic":0,"type":"boolean","defaultValue":"true","comment":"Forces mongod to validate all requests from clients upon
receipt."},{"scope":"Global","possibleValues":"false","name":"cpu","value":null,"isDynamic":0,"type":"boolean","defaultValue":"true","comment":"Forces mongod to report the percentage of the last interval spent in write-lock."}],"createdDate":"2014-04-03 17:58:08"}] }}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

4.7 Get Parameter Group by User Id for Paging

Http Method	URL	Description
GET	/NoSQL/api/v1/ NoSQLparametergroup?userId=<userId >&startIndex=<startIndex>&endIndex=<endIndex>	To get NoSQL parameter group by user ID

Query Parameters:

Filter Name	Type	Description	Required
userID	Int	Unique user ID of the user	Yes

startIndex	Int	Start index of paging	Yes
endIndex	Int	End index of paging	Yes

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlparametergroup?userId=2&startIndex=1&endIndex=2" -X GET -H "CL-AUTH-TOKEN:d4e129e6-fb83-4415-ba85-536eea1344a9"

Example Response:

{"success":true,"errors":{},"response":{"parameterGroups":[{"id":10,"description":"acfhas","name":"ghgsacas","engineName":"MongoDB-2.2.0","parameters":[{"scope":"Global","possibleValues":"false","name":"fork","value":null,"isDynamic":0,"comment":"Enables a daemon mode for mongod and allows you to run the database as a conventional server","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"quiet","value":null,"isDynamic":0,"comment":"This disables all but the most critical entries in output/log file","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"journal","value":null,"isDynamic":0,"comment":"Ensures single instance write-durability","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"nounixsocket","value":null,"isDynamic":0,"comment":"Disables the UNIX Socket","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"auth","value":null,"isDynamic":0,"comment":"Enables the authentication system within MongoDB","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"verbose","value":null,"isDynamic":0,"comment":"Enables a verbose logging mode that modifies mongod output and increases logging to include a greate","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"0,1,2,3,7","name":"diaglog","value":null,"isDynamic":0,"comment":"Enables diagnostic logging. Level 3 logs all read and write options.","defaultValue":"3","type":"Numeric"},{"scope":"Global","possibleValues":"false","name":"objcheck","value":null,"isDynamic":0,"comment":"Forces mongod to validate all requests from clients upon receipt.","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"cpu","value":null,"isDynamic":0,"comment":"Forces mongod to report the percentage of the last interval spent in write-lock.","defaultValue":"true","type":"boolean"}],"createdDate":"2014-04-19 02:37:22"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

5. NoSQL Parameter Group Family

5.1 Get All Parameter Group Family

Http Method	URL	Description
GET	/NoSQL/api/v1/ NoSQLparametergroupfamily	To get all NoSQL parameter group families

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlparametergroupfamily" -X GET -H "CL-AUTH-TOKEN:d4e129e6-fb83-4415-ba85-536eea1344a9"

Example Response:

{"success":true,"errors":{},"response":{"parameterGroupFamily":[{"id":1,"name":"MongoDB","engineName":"MongoDB-2.2.0","parameters":[{"scope":"Global","possibleValues":"false","name":"fork","isDynamic":0,"comment":"Enables a daemon mode for mongod and allows you to run the database as a conventional server","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"quiet","isDynamic":0,"comment":"This disables all but the most critical entries in output/log file","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"journal","isDynamic":0,"comment":"Ensures single instance write-

durability","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"nounixsocket","isDynamic":0,"comment":"Disables the UNIX Socket","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"auth","isDynamic":0,"comment":"Enables the authentication system within MongoDB","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"verbose","isDynamic":0,"comment":"Enables a verbose logging mode that modifies mongod output and increases logging to include a greate","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"0,1,2,3,7","name":"diaglog","isDynamic":0,"comment":"Enables diagnostic logging. Level 3 logs all read and write options.","defaultValue":"3","type":"Numeric"},{"scope":"Global","possibleValues":"false","name":"objcheck","isDynamic":0,"comment":"Forces mongod to validate all requests from clients upon receipt.","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"cpu","isDynamic":0,"comment":"Forces mongod to report the percentage of the last interval spent in write-lock.","defaultValue":"true","type":"boolean"}]},{"id":2,"name":"Couchbase","engineName":"Couchbase-2.2.0","parameters":[]}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

5.2 Get All Parameter Group Family by ID

Http Method	URL	Description
GET	/NoSQL/api/v1/ NoSQLparametergroupfamily/<id>	To get NoSQL parameter group families by ID

Query Parameters:

Filter Name	Type	Description	Required
ID	Int	Unique ID of the parameter group family	Yes

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlparametergroupfamily/1" -X GET -H "CL-AUTH-TOKEN:d4e129e6-fb83-4415-ba85-536eea1344a9"

Example Response:

{"success":true,"errors":{},"response":{"parameterGroupFamily":[{"id":1,"name":"MongoDB","engineName":"MongoDB-2.2.0","parameters":[{"scope":"Global","possibleValues":"false","name":"fork","isDynamic":0,"comment":"Enables a daemon mode for mongod and allows you to run the database as a conventional server","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"quiet","isDynamic":0,"comment":"This disables all but the most critical entries in output/log file","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"journal","isDynamic":0,"comment":"Ensures single instance write-durability","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"nounixsocket","isDynamic":0,"comment":"Disables the UNIX Socket","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"auth","isDynamic":0,"comment":"Enables the authentication system within MongoDB","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"verbose","isDynamic":0,"comment":"Enables a verbose logging mode that modifies mongod output and increases logging to include a greate","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"0,1,2,3,7","name":"diaglog","isDynamic":0,"comment":"Enables diagnostic logging. Level 3 logs all read and write options.","defaultValue":"3","type":"Numeric"},{"scope":"Global","possibleValues":"false","name":"objcheck","isDynamic":0,"comment":"Forces mongod to validate all requests from clients upon receipt.","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"cpu","isDynamic":0,"comment":"Forces mongod to report the percentage of the last interval spent in write-lock.","defaultValue":"true","type":"boolean"}]}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

5.3 Get All Parameter Group Family by Name

Http Method	URL	Description
GET	/NoSQL/api/v1/ NoSQLparametergroupfamily?name=<name>	To get NoSQL parameter group families by ID

Query Parameters:

Filter Name	Type	Description	Required
Name	String	Name of parameter group family	Yes

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlparametergroupfamily?name=MongoDB" -X GET -H "CL-AUTH-TOKEN:d4e129e6-fb83-4415-ba85-536eea1344a9"

Example Response:

{"success":true,"errors":{},"response":{"parameterGroupFamily":[{"id":1,"name":"MongoDB","engineName":"MongoDB-2.2.0","parameters":[{"scope":"Global","possibleValues":"false","name":"fork","isDynamic":0,"comment":"Enables a daemon mode for mongod and allows you to run the database as a conventional

server","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"quiet","isDynamic":0,"comment":"This disables all but the most critical entries in output/log
file","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"journal","isDynamic":0,"comment":"Ensures single instance write-durability","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"nounixsocket","isDynamic":0,"comment":"Disables the UNIX
Socket","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"auth","isDynamic":0,"comment":"Enables the authentication system within MongoDB","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"verbose","isDynamic":0,"comment":"Enables a verbose logging mode that modifies mongod output and increases logging to include a
greate","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"0,1,2,3,7","name":"diaglog","isDynamic":0,"comment":"Enables diagnostic logging. Level 3 logs all read and write
options.","defaultValue":"3","type":"Numeric"},{"scope":"Global","possibleValues":"false","name":"objcheck","isDynamic":0,"comment":"Forces mongod to validate all requests from clients upon
receipt.","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"cpu","isDynamic":0,"comment":"Forces mongod to report the percentage of the last interval spent in write-lock.","defaultValue":"true","type":"boolean"}]}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

5.4 Get All Parameter Group Family by Name and Version

Http Method	URL	Description
GET	/NoSQL/api/v1/ NoSQLparametergroupfamily?name=<name>&version=<version>	To get NoSQL parameter group families by ID

Query Parameters:

Filter Name	Type	Description	Required
Name	String	Name of parameter group family	Yes
Version	String	version of parameter group family	Yes

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/nosqlparametergroupfamily?name=MongoDB&version=2.2.0" -X GET -H "CL-AUTH-TOKEN:d4e129e6-fb83-4415-ba85-536eea1344a9"

Example Response:

{"success":true,"errors":{},"response":{"parameterGroupFamily":[{"id":1,"name":"MongoDB","engineName":"MongoDB-2.2.0","parameters":[{"scope":"Global","possibleValues":"false","name":"fork","isDynamic":0,"comment":"Enables a daemon mode for mongod and allows you to run the database as a conventional server","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"quiet","isDynamic":0,"comment":"This disables all but the most critical entries in output/log file","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"journal","isDynamic":0,"comment":"Ensures single instance write-durability","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"nounixsocket","isDynamic":0,"comment":"Disables the UNIX Socket","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","n

ame":"auth","isDynamic":0,"comment":"Enables the authentication system within MongoDB","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"verbose","isDynamic":0,"comment":"Enables a verbose logging mode that modifies mongod output and increases logging to include a greate","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"0,1,2,3,7","name":"diaglog","isDynamic":0,"comment":"Enables diagnostic logging. Level 3 logs all read and write
options.","defaultValue":"3","type":"Numeric"},{"scope":"Global","possibleValues":"false","name":"objcheck","isDynamic":0,"comment":"Forces mongod to validate all requests from clients upon
receipt.","defaultValue":"true","type":"boolean"},{"scope":"Global","possibleValues":"false","name":"cpu","isDynamic":0,"comment":"Forces mongod to report the percentage of the last interval spent in write-lock.","defaultValue":"true","type":"boolean"}]}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500, ...), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

6. Entity: NoSQL Subscription

6.1 Add Subscription

Http Method	URL	Description
POST	/NoSQL/api/v1/subscription	Creates the new subscription

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name "**add**"	Yes
Name	String	Unique subscription name	Yes
Price	String	Pricing of subscription in dollar ($)	Yes
priceType	String	Pricing type is monthly or yearly Expected values : Monthly=1 Yearly=2	Yes
Features	String	Different features of the subscription	Yes
storageSize	Integer	Size of storage	Yes
Multimode	Integer	Number of instances want to provision	Yes
iaasProviderId	Integer	Unique ID of iaas	Yes
imageInstanceTypeId	Integer	Unique ID of image instance type	Yes
engineId	Integer	Unique ID of Engine	Yes
availabilityZoneName	String	Name of zone in which instance in desired to be launched If targetcloud is VM pool then it not mandatory	Yes

Example Request:-

curl -i "http://localhost:8080/nosql/api/v1/subscription" -X POST -H "CL-AUTH-TOKEN:5fe422ca-c218-47b7-a107-e09bdc69678d" -d'{"add": {"subscription": {"name":

"SPnosqlSubscription","price": "10012$","priceType": "2","features": "NOSQLServiceonStandaloneVM
VMConfiguration:
Engine: MongoDB2.2.0
IaaSProvider: CloudStack
AvailabilityZone: 1
InstanceConfiguration:xsmall
NoofNodes: 1
Ext.Volume: 100GB
","storageSize": "15","iaasProviderId": "3","availabilityZoneName": "ContZone","engineId": "1","imageInstanceTypeId": "31","multiNode": "1"}}}'

Example Response:

{"success":true,"errors":{},"response":{"subscription":[{"ownerType":2,"priceType":2,"imageInstanceId":31,"id":4,"price":"10012$","createdBy":1,"nodes":1,"name":"SPnosqlSubscription","availabilityZone":"ContZone","storageSize":15,"feature":"NOSQLServiceonStandaloneVM
VMConfiguration:
Engine: MongoDB2.2.0
IaaSProvider: CloudStack
AvailabilityZone: 1
InstanceConfiguration:xsmall
NoofNodes: 1
Ext.Volume: 100GB
","engineId":1,"iaasProviderId":3,"targetCloudId":50,"createdDate":"2014-05-14 23:02:14"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), invalidParameterValue(1013), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)

6.2 Update NoSQL Subscription

Http Method	URL	Description
PUT	/NoSQL/api/v1/subscription/<id>	Updates the existing subscription

Request Parameters:

Attribute	Type	Description	Required
Id	Integer	Unique ID of existing subscription	Yes
Name	String	Unique subscription name	Yes
Price	String	Pricing of subscription in dollar ($)	Yes
priceType	String	Pricing type is monthly or yearly Expected values: Monthly=1 Yearly=2	Yes
Features	String	Different features of the subscription	Yes
storageSize	Integer	Size of storage	Yes
Multimode	Integer	Number of instances want to provision	Yes
iaasProviderId	Integer	Unique ID of iaas	Yes
imageInstanceTypeId	Integer	Unique ID of image instance type	Yes
engineId	Integer	Unique ID of engine	Yes
availabilityZoneName	String	Name of zone in which instance is desired to be	Yes

		launched If targetcloud is VM pool then it not mandatory	

Example Request:

curl -i "http://localhost:8080/nosql/api/v1/subscription/2" -X PUT -H "CL-AUTH-TOKEN:64d48eb6-70fa-406f-a713-5604dba2e90c" -d'{"name": "ABCDddd","price": "10$","priceType": "1","features": "NooooStandaloneVM VMConfiguration: Engine: MongoDB2.2.2IaaSProvider: HPCSAvailabilityZone: zone1InstanceConfiguration:standard.xsmall-1vCPU/1GBRAM/30GBHD-$1 NoofNodes: 10Ext.Volume: 100GB","storageSize": "20","iaasProviderId": "5","availabilityZoneName": "zone#2","engineId": "1","imageInstanceTypeId": "32","multiNode": "2"}'

Example Response: JSON

{"success":true,"errors":{},"response":{}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), invalidParameterValue(1013), entityDoesNotExist(1012), invalidAction(1001), invalidRequest(1000)

6.3 Delete Subscription

Http Method	URL	Description
DELETE	/NoSQL/api/v1/subscription/<id>	Deletes the existing subscription

Example Request:

curl -i "http://localhost:8080/nosql/api/v1/subscription/2" -X DELETE -H "CL-AUTH-TOKEN:64d48eb6-70fa-406f-a713-5604dba2e90c"

Response Format:

{"success":true,"errors":{},"response":{}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidParameterValue(1013), invalidRequest(1000)

6.4 Get Subscription by ID

Http Method	URL	Description
GET	/NoSQL/api/v1/subscription?id=<id>	To get specific subscription

Example Request:

curl -i "http://localhost:8080/nosql/api/v1/subscription/3" -X GET -H "CL-AUTH-TOKEN:64d48eb6-70fa-406f-a713-5604dba2e90c"

Example Response:

{"success":true,"errors":{},"response":{"subscription":[{"priceType":1,"imageInstanceId":32,"id":3,"price":"100$","createdBy":1,"nodes":1,"name":"test_nosql","availabilityZone":"zone#2","storageSize":10,"feature":"NOSQLServiceonStandaloneVMVMConfiguration: Engine: MongoDB 2.2.0IaaSProvider: HPCSAvailabilityZone: zone2InstanceConfiguration:standard.xsmall-1vCPU/1GBRAM/30GBHD-$1NoofNodes: 1Ext.Volume: 100GB ","engineId":1,"iaasProviderId":5,"targetCloudId":34,"createdDate":"2014-05-10 02:35:21"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), entityDoesNotExist(1012), invalidRequest(1000)

6.5 Get all Subscription

Http Method	URL	Description
GET	/NoSQL/api/v1/subscription	To get all NoSQL subscription

Example Request:

curl -i "http://localhost:8080/nosql/api/v1/subscription" -X GET -H "CL-AUTH-TOKEN:64d48eb6-70fa-406f-a713-5604dba2e90c"

Example Response:

{"success":true,"errors":{},"response":{"subscription":[{"priceType":1,"imageInstanceId":32,"id":2,"price":"10$","createdBy":1,"nodes":2,"name":"ABCDze":20,"feature":"NooooStandaloneVMVMConfiguration: Engine: MongoDB2.2.2IaaSProvider: HPCSAvailabilityZone: zone1InstanceConfigur1
NoofNodes: 10Ext.Volume: 100GB","engineId":1,"iaasProviderId":5,"targetCloudId":34,"createdDate":"2014-05-10 02:32:06"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), invalidRequest(1000)

6.6 Get all Subscription by user

Http Method	URL	Description
GET	/NoSQL/api/v1/subscription?userId=<id>	To get all SQL subscription by user

Example Request:

curl -i "http://localhost:8080/nosql/api/v1/subscription?userId=2" -X GET -H "CL-AUTH-TOKEN:64d48eb6-70fa-406f-a713-5604dba2e90c"

Example Response:

{"success":true,"errors":{},"response":{"subscription":[{"priceType":1,"imageInstanceId":32,"id":3,"price":"100$","createdBy":1,"nodes":1,"name":"test_nosql","availabilityZone":"zone#2","storageSize":10,"feature":"NOSQLServiceonStandaloneVMVMConfiguration: Engine: MongoDB 2.2.0IaaSProvider: HPCSAvailabilityZone: zone2InstanceConfiguration:standard.xsmall-1vCPU/1GBRAM/30GBHD-$1NoofNodes: 1Ext.Volume: 100GB ","engineId":1,"iaasProviderId":5,"targetCloudId":34,"createdDate":"2014-05-10 02:35:21"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413)
Custom Error Response Code(s):	queryFailed(1009), invalidRequest(1000)

7. Entity: MongoDB Management Service (MMS) Instance

7.1 Create MMS Instance

Http Method	URL	Description
POST	/mms/instance	Creates a new MMS instance

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name “**create**”	Yes
groupId	String	Unique MMS group name	Yes
apiKey	String	Unique MMS api key	Yes

targetCloudId	Integer	Unique ID of created target cloud
		Ex. "target Cloud":
		{
		"id": "1"
		}
			Yes

instanceTypeId	Integer	Unique ID of instance type
		Ex. "instanceType":
		{
		"id": "1"
		}
			Yes

availabilityZone	String	Name of zone in which launch of instance is desired	Yes

Example Request:

$ curl -i http://localhost:8080/nosql/api/v1/mms/instance -X POST -H "CL-AUTH-TOKEN:f223681a-9fb1-495e-b66a-56906c8f9c67" -d'{"create":{"mmsInstance":{"groupId":"SWERP12","apiKey":"09c3221cdbec74a62f1f248c1ba47e79","targetCloud":{"id":17},"availabilityZone":"f9a1d2eb-c49e-4b87-ac9b-9c47e2d912a3","instanceType":{"id":11}},"isProvision":true}}'

Example Response:

{"success":true,"errors":{},"response":{"mmsInstance":[{"serviceaccessgroup":{"id":4414,"name":"default_mms_183"},"id":5,"groupName":"SWERP12","targetcloud":{"id":17,"name":"CLOUDCENTRAL_TARGET_CLOUD"},"createdBy":2,"status":2,"nodes":1,"availabilityZone":"f9a1

d2eb-c49e-4b87-ac9b-9c47e2d912a3","isInProgress":2,"createdDate":"2014-05-31 17:45:54","instancetype":{"id":11,"instanceTypeId":"2567bd68-f186-4f09-9f99-4fd75b1f756b","name":"Medium Compute"}}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)

7.2 Delete MMS Instance

Http Method	URL	Description
DELETE	/NoSQL/api/v1/mms/instance	Deletes the existing NoSQL MMS instance

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/mms/instance/2" -X DELETE -H "CL-AUTH-TOKEN: f223681a-9fb1-495e-b66a-56906c8f9c67"

Response Format:

{"success":true,"errors":{},"response":{}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)

7.3 Get All MMS Instances

Http Method	URL	Description
GET	/NoSQL/api/v1/mms/instance	To Get all NoSQL MMS instance

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/mms/instance" -X GET -H "CL-AUTH-TOKEN:f223681a-9fb1-495e-b66a-56906c8f9c67"

Response Format:

{"success":true,"errors":{},"response":{"mmsInstance":[{"serviceaccessgroup":{"id":4413,"name":"default_mms_173"},"id":4,"groupName":"swuioiuuu","serviceTag":16,"targetcloud":{"id":17,"name":"CLOUDCENTRAL_TARGET_CLOUD"},"createdBy":2,"status":1,"nodes":1,"availabilityZone":"f9a1d2eb-c49e-4b87-ac9b-9c47e2d912a3","isInProgress":2,"createdDate":"2014-

05-31 14:29:14","instancetype":{"id":11,"instanceTypeId":"2567bd68-f186-4f09-9f99-4fd75b1f756b","name":"Medium Compute"}}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)

7.4 Get MMS Instance By ID

Http Method	URL	Description
GET	/NoSQL/api/v1/mms/instance/<id>	To Get NoSQL MMS instance by instance ID

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/mms/instance?id=4" -X GET -H "CL-AUTH-TOKEN:f223681a-9fb1-495e-b66a-56906c8f9c67"

Response Format:

{"success":true,"errors":{},"response":{"mmsInstance":[{"serviceaccessgroup":{"id":4413,"na

me":"default_mms_173"},"id":4,"groupName":"swuioiuuu","serviceTag":16,"targetcloud":{"id":17,"name":"CLOUDCENTRAL_TARGET_CLOUD"},"createdBy":2,"status":1,"nodes":1,"availabilityZone":"f9a1d2eb-c49e-4b87-ac9b-9c47e2d912a3","isInProgress":2,"createdDate":"2014-05-31 14:29:14","instancetype":{"id":11,"instanceTypeId":"2567bd68-f186-4f09-9f99-4fd75b1f756b","name":"Medium Compute"}}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)

7.5 Provision MMS Instance

Http Method	URL	Description
POST	/NoSQL/api/v1/mms/instance	Starts provisioning of the existing MMS instance

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name "**provision**"	Yes
ID	Integer	Unique ID of the existing instance	Yes

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/mms/instance" -X POST -H "CL-AUTH-TOKEN:7f4b8754-3442-4569-bccf-ab3a40451106" -d'{"provision": { "mmsInstance": { "id": "1"}}}'

Example Response:

{"success":true,"errors":{},"response":{}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), invalidRequest(1000)

7.6 Terminate MMS Instance

Http Method	URL	Description
POST	/NoSQL/api/v1/mms/instance	Terminates the existing MMS instance

Request Parameters:

Attribute	Type	Description	Required

<action>	String	Action name "**terminate**"	Yes
ID	Integer	Unique ID of the existing instance	Yes

Example Request:

$ curl -i "http:// localhost:8080/nosql/api/v1/mms/instance" -X POST -H "CL-AUTH-TOKEN:af169236-ed07-41a9-bbaf-47f779df01c3" -d'{"terminate": {"mmsInstance": {"id": "1"}}}'

Example Response: JSON

{"success":true,"errors":{},"response":{}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), invalidRequest(1000)

7.7 Cancel MMS Instance

Http Method	URL	Description
POST	/NoSQL/api/v1/mms/instance	Cancel provisioning of the existing MMS instance

Request Parameters:

Attribute	Type	Description	Required
<action>	String	Action name “**cancel**”	Yes
ID	Integer	Unique ID of the existing instance	Yes

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/mms/instance" -X POST -H "CL-AUTH-TOKEN:7f4b8754-3442-4569-bccf-ab3a40451106" -d'{"cancel": { "mmsInstance": { "id": "1"}}}'

Example Response:

{"success":true,"errors":{},"response":{}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), invalidRequest(1000)

7.8 Get MMS Access Details By ID

Http Method	URL	Description
GET	/NoSQL/api/v1/ mms/accessdetails /<id>	To get NoSQL MMS access details by instance ID

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/mms/accessdetails?id=4" -X GET -H "CL-AUTH-TOKEN:f223681a-9fb1-495e-b66a-56906c8f9c67"

Example Response :

{"success":true,"errors":{},"response":{"accessdetails":[{"port":0,"id":69,"hostType":28,"isMultiAzNode":false,"name":"111.125.178.77","userName":"root","privateDnsName":"cumulogic-10-swuioiuuu-2072","publicDnsName":"111.125.178.77","createdDate":"2014-05-31 14:30:10","password":"testuser","haStatus":1,"privateIpAddress":"192.168.2.109"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)

7.9 Get MMS Events By ID

Http Method	URL	Description
GET	/NoSQL/api/v1/ mms/events /<id>	To get NoSQL MMS events by instance ID

Example Request:
$ curl -i "http://localhost:8080/nosql/api/v1/mms/events?id=4" -X GET -H "CL-AUTH-TOKEN:f223681a-9fb1-495e-b66a-56906c8f9c67"

Example Response:

{"success": true,"errors": {},"response": {"events": [{"message": "Create instance task started ","id": 2495, "createdBy": "2014-05-31 14:29:36","type": "INFO" }, {"message": "Create instance job started","id": 2494,"createdBy": "2014-05-31 14:29:36","type": "INFO"},{"message": "Instance network configuration started for Host 192.168.2.109", "id": 2497,"createdBy": "2014-05-31 14:30:10","type": "INFO"}, {"message": "Create instance task completed for Host 192.168.2.109","id": 2496,"createdBy": "2014-05-31 14:30:10","type": "INFO"},{"message": "Validate instance connection started for Host 111.125.178.77","id": 2499,"createdBy": "2014-05-31 14:31:29","type": "INFO"},{
 "message": "Instance network configuration completed for Host 111.125.178.77","id": 2498,"createdBy": "2014-05-31 14:31:29","type": "INFO"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)

8. Entity: NoSQL Events

8.1 Get NoSQL Events By ID

Http Method	URL	Description
GET	NoSQL/api/v1/instance/events/<id>	To get NoSQL events by instance ID

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/instance/events?id=8" -X GET -H "CL-AUTH-TOKEN:ef269509-3e90-4308-9847-59e39b4f1e4e"

Example Response:

{"success":true,"errors":{},"response":{"events":[{"message":"Create instance task started ","id":1641,"createdBy":"2014-05-31 02:14:30","type":"Create instance task started "},{"message":"Create instance job started","id":1640,"createdBy":"2014-05-31 02:14:30","type":"Create instance job started"},{"message":"Instance network configuration started for Host 192.168.2.24","id":1643,"createdBy":"2014-05-31 02:15:03","type":"Instance network configuration started for Host 192.168.2.24"},{"message":"Create instance task completed for Host 192.168.2.24","id":1642,"createdBy":"2014-05-31 02:15:03","type":"Create instance task completed for Host 192.168.2.24"},{"message":"Validate instance connection started for Host 111.125.178.84","id":1647,"createdBy":"2014-05-31 02:16:14","type":"Validate instance connection started for Host 111.125.178.84"},{"message":"Instance network configuration completed for Host 111.125.178.84","id":1646,"createdBy":"2014-05-31 02:16:14","type":"Instance network configuration completed for Host 111.125.178.84]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404),

	serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)

9. Entity: NoSQL Monitoring

9.1 Get NoSQL Monitoring Charts

Http Method	URL	Description
GET	NoSQL/api/v1/ instance/monitoring/ <id>	To get monitoring charts by instance ID

Example Request:

$ curl -i "http://localhost:8080/nosql/api/v1/instance/monitoring?id=8" -X GET -H "CL-AUTH-TOKEN:ef269509-3e90-4308-9847-59e39b4f1e4e"

Example Response :

{"success":true,"errors":{},"response":{"monitoring":[{"metricData":[[1401528023000,2],[1401528053000,0],[1401528083000,0],[1401528113000,0],[1401528143000,0],[1401528173000,0],[1401528203000,0],[1401528233000,0],[1401528263000,0],[1401528293000,1],[1401528323000,0],[1401528353000,0],[1401528383000,0],[1401528413000,0],[1401528443000,0],[1401528473000,0],[1401528504000,0],[1401528534000,0],[1401528564000,0],[1401528594000,0],[1401528624000,0]],"id":8,"metricLabel":"[CPU]User%","ownerType":2,"instanceLabel":"111.125.178.84"}"id":8,"metricLabel":"[DSK]WriteKBTot","ownerType":2,"instanceLabel":"111.125.178.84"}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)

10. Entity: Access Details

10.1 Get Access Details of Provisioning Instance

Http Method	URL	Description
GET	/NoSQL/api/v1/ mms/accessdetails /<id>	To get access details of NoSQL instance by instance ID

Example Request:

$ curl -i "http:// localhost:8080/nosql/api/v1/instance/accessdetails?id=8" -X GET -H "CL-AUTH-TOKEN:ef269509-3e90-4308-9847-59e39b4f1e4e"

Example Response :

{"success":true,"errors":{},"response":{"accessdetails":[{"port":27017,"hostType":7,"isMultiAzNode":false,"privateDnsName":"cumulogic-3-fdgfhf-9025","password":null,"isDnsConfigured":true,"privateIpAddress":"192.168.2.24","id":47,"na

me":"111.125.178.84","userName":"root","publicDnsName":"111.125.178.84","createdDate":"2014-05-31 02:15:03","haStatus":1}]}}

Response Code(s):

Normal Response Code(s):	200
Error Response Code(s):	computeFault (400, 500), serviceUnavailable (503), unauthorized (401), forbidden (403), badRequest (400), badMethod (405), overLimit (413), itemNotFound (404), serverCapacityUnavailable (503)
Custom Error Response Code(s):	queryFailed(1009), missingParameter(1004), invalidAction(1001), entityAlreadyExist(1011), invalidRequest(1000)