

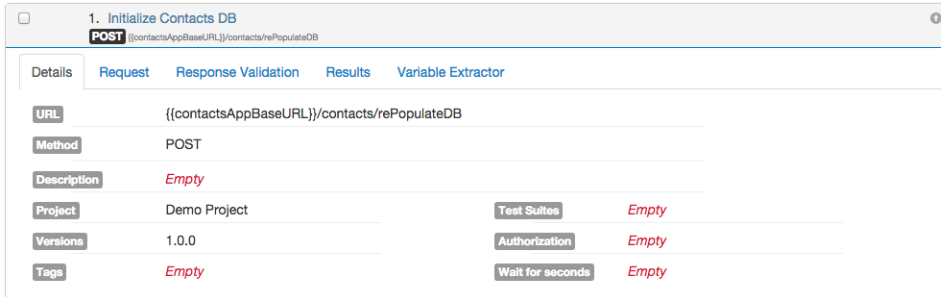
# Setup and Tear Down for Test Cases

In this article, we will see how we perform setup and tear down operation for test cases in vREST.

Here is a video tutorial for the same:

The setup or tear down method is simply yet another test case in vREST. The idea is, you need to implement some kind of logic over REST API to initialize your test application. This can be done in two ways:

1. Implement a REST API endpoint in the test application itself, which is open only in test environment and which will insert the initialization data upon invocation of the API.



Note: You may also create a separate application which loads data in your database and provides you an API for that.

2. Create another application, which provide REST APIs to execute commands on the test machine in which test application's data resides. Now with the help of this application, we can insert database dumps through commands and these commands can be invoked via REST APIs.

We have created an external module named as vutil for this purpose. For installation, please visit [Setup and Install vutil](#).

vutil provides a REST API to execute commands on the test machine in which test application's data resides. Now with the help of this API, we can insert database dumps through commands and these commands can be invoked via REST APIs.

This API execute commands on machines on which this utility is installed.

## API Endpoint:

The API format is

```
POST {{vutilBaseURL}}/execute/command
```

Here {{vutilBaseURL}} is the base URL of the vutil server.

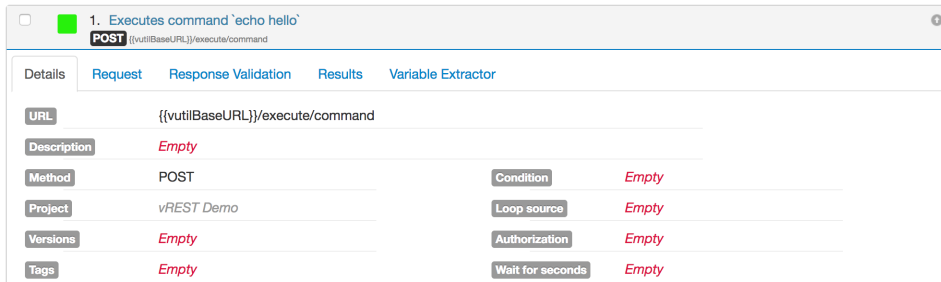
and this API accepts a single body parameter "command" and its value is the command which you want to execute on your system.

## How to use this API

Assuming that vutil is running on port 4080 and vutil base URL is "<http://localhost:4080>"

Let us try to execute a simple command `echo hello` via this utility. Simply create a test case with URL "<http://localhost:4080>".

Your test case might look like:



and provide a request parameter "command" as below screenshot describes:

Name	Description	Field Type	Parameter Type	Value
command		string	body	echo hello

and provide some assertions to validate whether the command executed successfully or not.

#	Source	Property	Comparison	Expected Value
1	Status Code		Equal to number	200
2	Text Body		Call Default Validator	Show expected body

and finally the expected body for the test case.

```

1- {
2  "pid": "{{*}}",
3  "output": "hello\n"
4  }

```

Now, if we execute this test case. Then the above REST API executed the command `echo hello` on the remote machine and returned the command output as response as shown in below figure.

**Test Results** Pre Hooks Post Hooks

Result : **Passed** Test Run : 7:23:59.015 pm, Apr 15, 2017

**Execution time details :**

Details Request Headers Response Headers

**POST** http://localhost:4080/execute/command  
 Status code : 200 Response Time : 23 ms

**Actual request body :**  
 command=echo%20hello

**Actual response body :**

```

1- {
2  "pid": 23182,
3  "output": "hello\n"
4  }

```

Hope you got the idea.

Now, let us take a more practical example of using this utility. Suppose, I want to restore my MongoDB database before I execute my test suite. And my MongoDB dump is stored at `{{dataDir}}/xyz-ts-dump` directory. Then I can write a separate test case in vREST like below:

2. Restore database for Test Suite `XYZ`
POST {{vutilBaseUrl}}/execute/command

Details
Request   Response Validation   Results   Variable Extractor

URL
{{vutilBaseUrl}}/execute/command

Description
Empty

Method
POST
Condition
Empty

Project
vREST Demo
Loop source
Empty

Versions
Empty
Authorization
Empty

Tags
Empty
Wait for seconds
Empty

2. Restore database for Test Suite `XYZ`
POST {{vutilBaseUrl}}/execute/command

Details
Request   Response Validation   Results   Variable Extractor

Parameters
Raw Body   Headers

New
Delete
Copy
Paste
Bulk Operation
Generate Test Cases

Name	Description	Field Type	Parameter Type	Value
<input type="checkbox"/>	command	string	body	mongorestore --db dbname --drop {{dumpDir}}/xyz-ts-dump
<input type="checkbox"/>				

Similarly other details can be provided as described in previous example.

You can even execute some custom script via this API to initialize your Test Suite data depending upon your context.