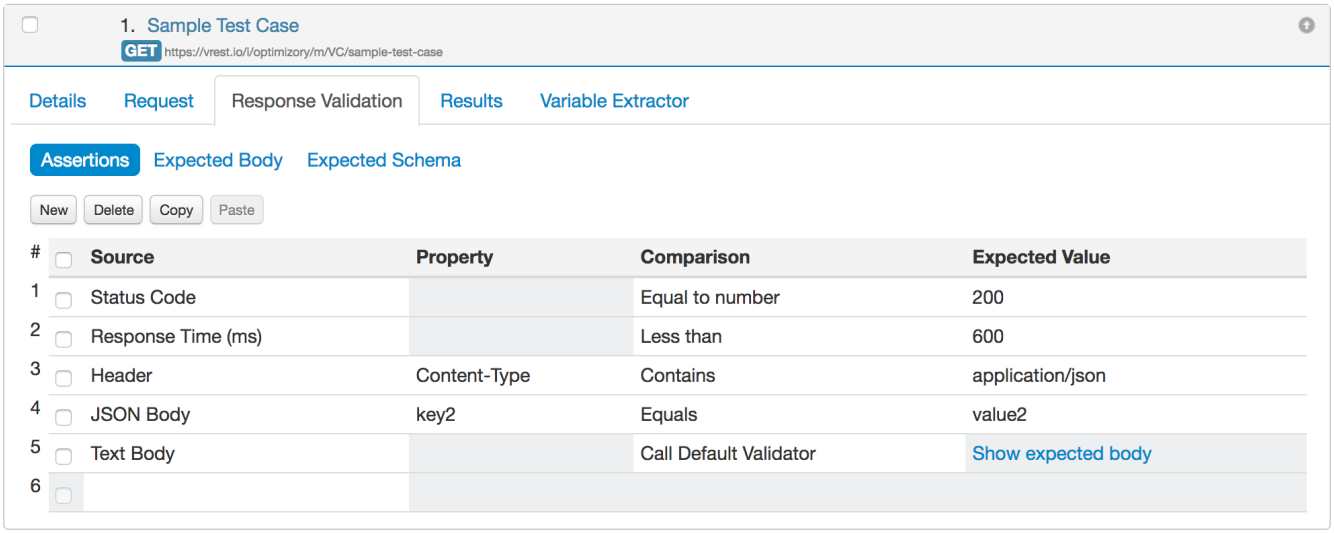
Response Validation

In vREST, response validation is done with the help of various assertions and/or response validators.

Here is the video tutorial for the same:

With the help of assertions and response validators, you can validate various response attributes like

- Status Code,
- Response Headers,
    - Useful to validate the response header's values.
- Response Time,
    - Useful to validate the API's performance. If an API is not returning response in pre-defined time period then APIs implementation can be improved.
- Response Content (Text Body or JSON Body or XML Body)
    - here Text body assertions can be applied to any type of response text content.



The above screenshot gives an idea of how vREST handles response validation. The above assertions validates the following:

1. Validates that the response status code is equal to 200.
2. Validates that the response time of the API is less than 600 ms.
3. Validates that response header "Content-Type" contains the substring "application/json".
4. Validates that JSON Body property `key2` value is equal to `value2`.
5. Validates that response body matches the pre-defined expected body.

In the above image, the last "Text Body" assertion is of prime importance. The last assertion basically calls "Default Validator" (vREST's in-built response validator) to validate your response body. In this case, you need to set the expected response body of the test case in "Expected Body" sub-tab. The default validator is very simple to use and provides you a powerful way to validate your test case responses.

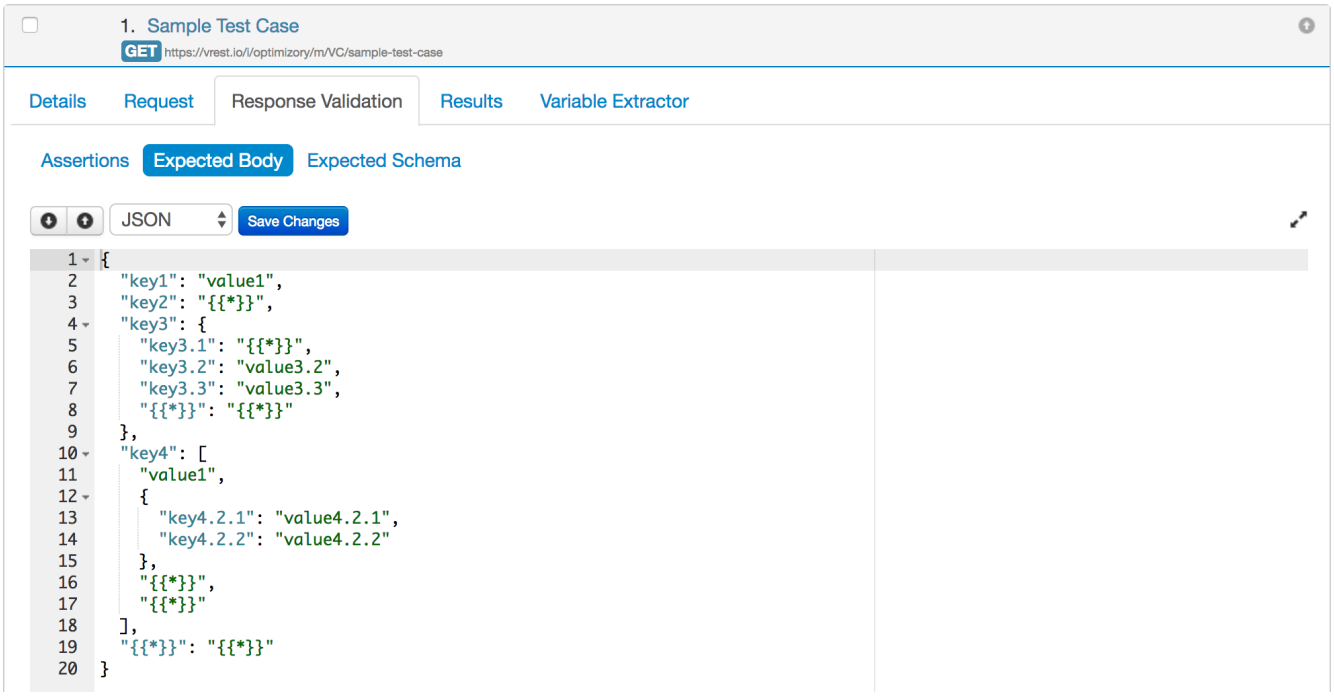
Suppose your test case responds the following JSON response:

```
{
    "key1": "value1",
    "key2": "value2",
    "key3": {
        "key3.1": "value3.1",
        "key3.2": "value3.2",
        "key3.3": "value3.3",
        "key3.4": "value3.4",
                "key3.5": "value3.5"
    },
    "key4": [
        "value1",
        {
            "key4.2.1": "value4.2.1",
            "key4.2.2": "value4.2.2"
        },
        "value3",
        "value4"
    ],
    "key5": "value5",
        "key6": "value6"
}
```

And you want to validate the following:

1. Validate whether the values of key1, key3.2, key3.3 are same as expected.
2. Validate whether first two values of the array "key4" are same as expected.
3. Validate whether the keys key2, key3.1 exists in the response or not. We only want to check the existence of keys, not their values.

Then you may write your expected body like this:



Note:

- "{{*}}" is a special variable,
    - If used as a value in a JSON response, the value is simply ignored while comparing it with actual response body. The value can be a simple object or can be a large nested object.
    - If used as key/value pair in a JSON response, then rest of the key-value pairs are simply ignored while comparing the expected body with actual response body.

For more information on assertions and validators, please follow the links below:

1. TC Assertions sub tab
2. Response Validators