{REST:API}

iRest: Rest API Simulator

END USER MANUAL

Table of Contents

iRest – User Manual 2

Login Section 2

MOCK LIST Section 2

- Enable/Disable a Mock 3
- Delete a Mock 3
- Get the Mock URL 3
- Create a Mock 3
- Edit a Mock 6

Simulate a Default GET API 7

Simulate a Default POST API 8

Simulate a SAVE POST API 9

Simulate a SAVE PUT API 10

Simulate a SAVE PATCH API 11

Simulate a SAVE DELETE API 12

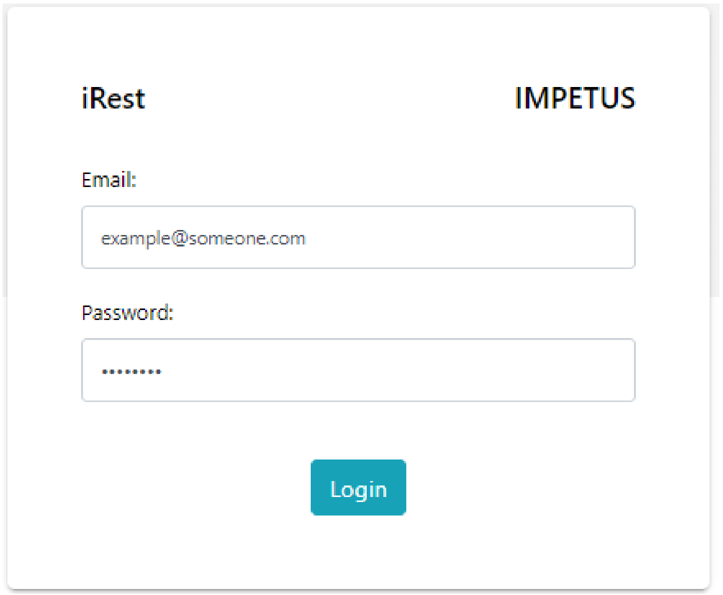
Simulate a EXISTING API 13

IREST – USER MANUAL

This manual consists all the functionalities a user can do within iRest.

LOGIN SECTION

Every User must need to be authenticated from the login screen.



Email – Your Email address for iRest.

Password – Your Password for the iRest.

MOCK LIST SECTION

Mock List section consists all your created mocks in one place. This section consists various functionality as mentioned below.

1. Enable/Disable a Mock
2. Delete a Mock
3. Get the Mock URL
4. Create a new Mock
5. Edit an existing Mock

Enable/Disable a Mock

1. Login into iRest.
2. Go to Mock List Section.
3. Choose a Mock for which you want to change the status.
4. Click on the Play/Pause Icon to Enable/Disable respectively under the **Start/Stop Column**.
5. Verify your API.

Delete a Mock

1. Login into iRest.
2. Go to Mock List Section.
3. Choose a Mock for which you want to delete permanently.
4. Under the **Actions** column, Click on the delete Icon to delete your mock permanently.

Get the Mock URL

1. Login into iRest.
2. Go to Mock List Section.
3. Choose a Mock for which you want to copy the URL.
4. Under the URL column, Click on the copy Icon below the URL. It will copy your URL to your clipboard. Paste it wherever you need it.

Create a Mock

1. Login into iRest.
2. Go to Mock List Page.
3. Click on the Add Icon on bottom right corner on the page.
4. On the next page there are various fields as listed below.
5. Fill the form & click on submit to create a mock.

Create Mock Fields

- **Base URL** – A predefined field consists of base URL of the mock.
- **Project Name** – A searchable input field for creating or choosing an existing project.
    a. To create a new project, type in the value and click on add icon to create a new project.
    b. For existing project, type in the search value in input box and choose an existing project from the dropdown.

- **Endpoint** – An input field for the endpoint of the mock URL. Type in your suitable endpoint in this field.

  The complete URL is the combination of above fields: Base URL + Project Name + Endpoint. For eg.

  http://localhost/api/rest *(***Base URL***)* + ***test-project-name*** *(***Project Name***)* + ***test-endpoint*** **(Endpoint)**

  The complete Mock URL looks like the below one:

  http://localhost/api/rest/test-project-name/test-endpoint

- **Mock Type** - Mock Type could be any of three mentioned below. Choose one of them as per your requirements.
  - **Default**: Return the JSON/Plain-Text Data in Response with certain optional advanced options such as delay, dynamic records, etc.
  - **Save**: Mock APIs with CRUD Operations such as create, modify and delete the record.
  - **Existing**: A Listing API with all the records stored during the CRUD Operation in 2nd type (Save) of API.

- **Request Method** - The HTTP request method of API call among **GET/ POST/ PUT/ PATCH/ DELETE** http methods. Choose an HTTP method which you want to use in your project.

  In the basic scenario, a user want to retrieve the data only. In such scenarios, only GET/ POST methods are sufficient to receive the data. You can simply choose mock type as '*Default*' and choose a request method from GET/POST.

  If the complex scenario such as CRUD requirements, you can simply choose mock type '*Save*' and Choose a request method from POST/PUT/PATCH/DELETE.

  There is another specific scenario where you want to retrieve all your stored records at once during implementing the POST call to save the data. For such scenarios, you can create an API with mock type '*Existing*'. These scenario helps you to create a record list API.

- **Response Status Code** - The HTTP response status code from the API. It could be any one of these status codes: **200/ 400/ 401/ 500/ 502**. Please choose one that suite your requirements.

- **Content Encoding** - Content Encoding of data. Default value sets to 'UTF-8'.

  Mock Name – Name of the mock. Type in the value to set the name of your mock.

- **Mock Status** - A checkbox for toggle the active state of mock. If checkbox is ticked, mock will be enabled. Otherwise you need to update the mock status from listing page.

Additional Fields

The below additional fields scenario specific. Depends upon the type of mock you choose, these field will be available as per the specific use case.

Scenario 1

For mock type "***Default***", these additional fields needed to be filled along with previous fields.

- **Content Type** - A list of all the content type which is expected from the Mock API response. Choose an option between these two fields: *Text/Plain, Application/JSON.*
- **Response Body** - Write down the response body data which you are expecting as your API response.
  (OR)
- **Upload Response Body** – Click on upload button and attach your large JSON response body you are expecting from your API. Once you upload the file, all the content is populated into the response body input.
- **Advanced Response** - Optional, this contains some optional fields user can choose if required.
  a. **Delay** – If you want to your response to be delayed by a few seconds, you can use this toggle button. Once you enable it, it will open an input against it. Enter the value in seconds you want to delay. 60 seconds is the maximum limit for delay.
  b. **Dynamic Response** – If you want to give dynamic data that will increase your data records limits up to an extent. You can enable the dynamic option and use it in your API. This Section consists the field mentioned below.
     i. **Key** - A Key from the JSON response body section on which dynamic operation is operated.
     ii. **Dropdown for Type of Key Value** - Dropdown consists 2 options, *random* & *specific*.
        1. Choose **Random** type of key value if you need dummy data to be prefilled from the API end.
        2. If you choose **Specific** type of key value, an input field should be added for key value. You need to specify the value of the key on yourself. The value will be repeated in the *Key* within your response body which you mentioned in last step.
     iii. Dropdown for quantity of dynamic records - Dropdown consists 2 options, count & size.
     1. Choose **Count** Option and write the number of records you want within input field. Max count limits to 1000000 (10 Lakhs). <u>Records above 100000 (1 Lakh), needs streaming to be enabled from client end.</u>
     2. Choose **Size** Option, and write the size of records in KB within input field. The Max size limit is 20480 KB (20 MB). <u>Records above 1024 KB (1 MB), needs streaming to be enabled from client end.</u>

Scenario 2

For mock type “***Save***”, these additional fields needed to be filled along with previous fields.

- **Advanced Response** - Optional, this contains some optional fields user can choose if required.
  a. **Delay** – If you want to your response to be delayed by a few seconds, you can use this toggle button. Once you enable it, it will open an input against it. Enter the value in seconds you want to delay. 60 seconds is the maximum limit for delay.

Scenario 3

For mock type “***Existing***”, these additional fields needed to be filled along with previous fields.

- **Advanced Response** - Optional, this contains some optional fields user can choose if required.
  a) **Delay** – If you want to your response to be delayed by a few seconds, you can use this toggle button. Once you enable it, it will open an input against it. Enter the value in seconds you want to delay. 60 seconds is the maximum limit for delay.

Optional Fields

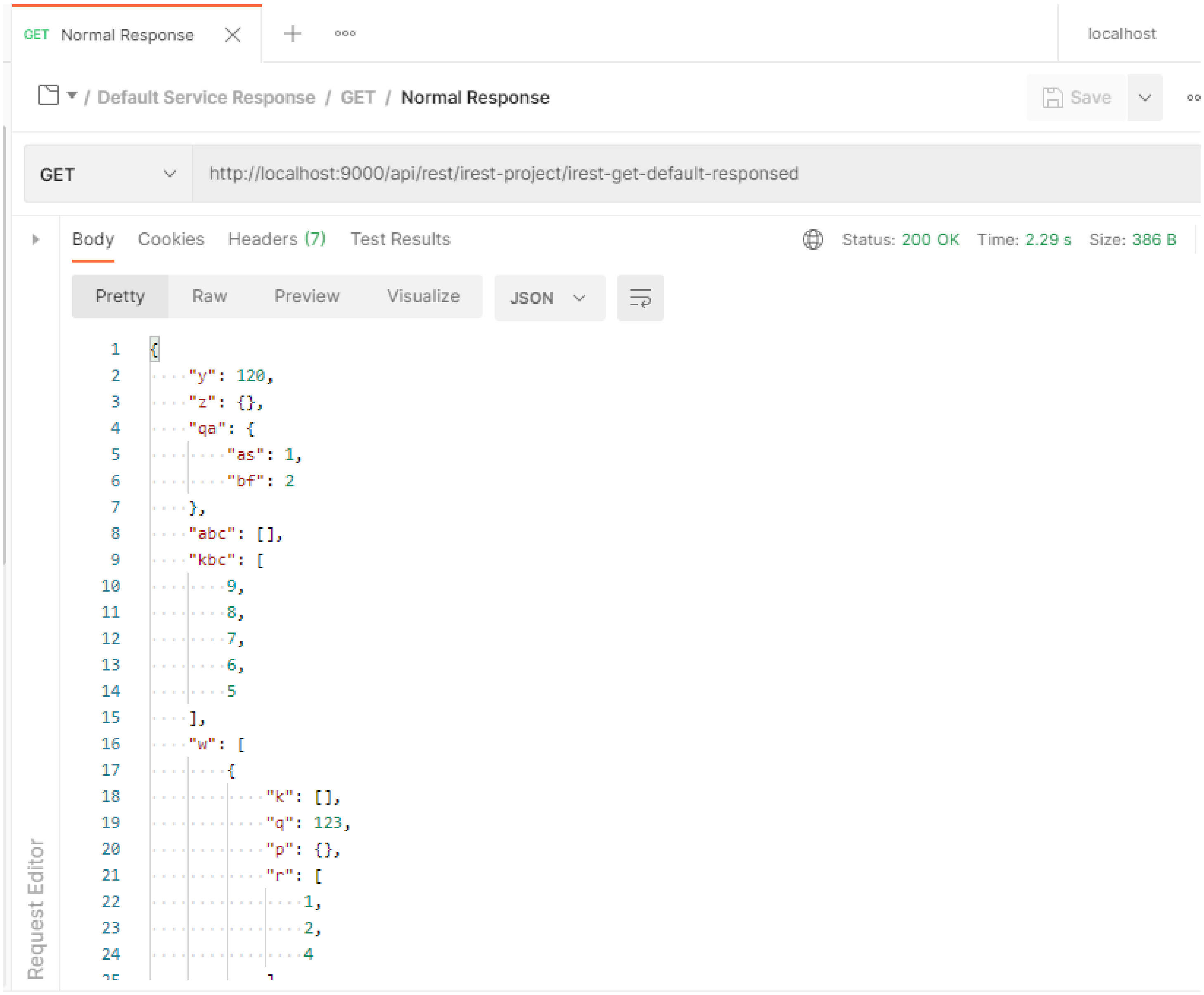
- **Advanced Options** - Optional for user, this contains some optional fields user can choose.
  - **Headers**: A combination of key/value pair of response headers. You can fill any amount of custom headers. These custom headers will be added to the API response.
  - **Params**: A combination of key/value pair of request parameters. A user can fill any amount of custom parameters. There is not limit for the number of request parameters. You can use the request headers with the request URL of the mock.

Edit a Mock

1. Login into iRest.
2. Go to Mock List Page.
3. Choose a Mock for which you want to edit a mock.
4. Click on the Edit Icon from the record.
5. On the next page edit the fields on the page & click on submit to update the mock. Check the ***create mock sectio***n for the each fields detail.

Annexure 1.

SIMULATE A DEFAULT GET API



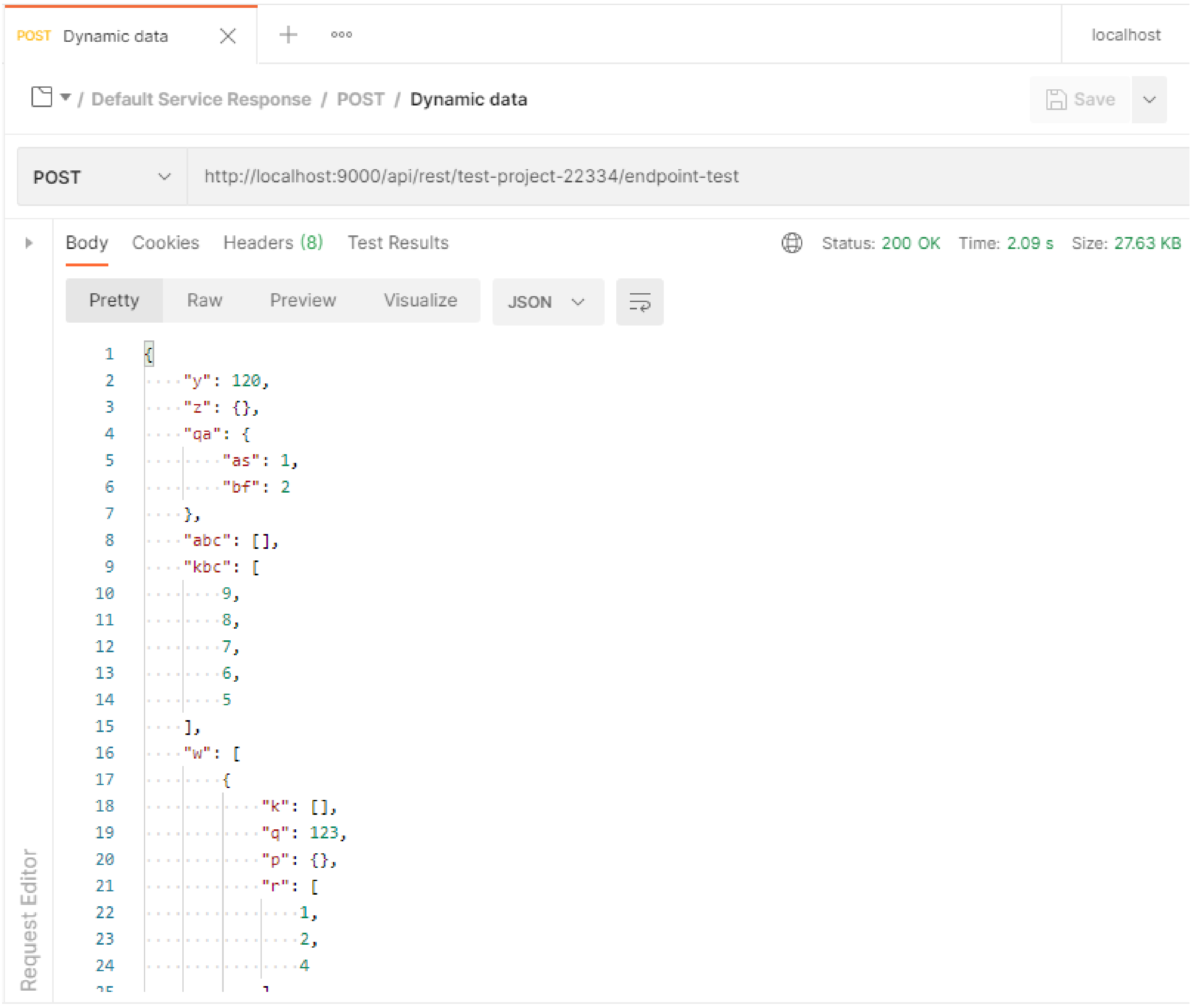
GET http://localhost:9000/api/rest/irest-project/irest-get-default-responsed

Status: 200 OK Time: 2.29 s Size: 386 B

```
{
    "y": 120,
    "z": {},
    "qa": {
        "as": 1,
        "bf": 2
    },
    "abc": [],
    "kbc": [
        9,
        8,
        7,
        6,
        5
    ],
    "w": [
        {
            "k": [],
            "q": 123,
            "p": {},
            "r": [
                1,
                2,
                4
```

Annexure 2.

SIMULATE A DEFAULT POST API



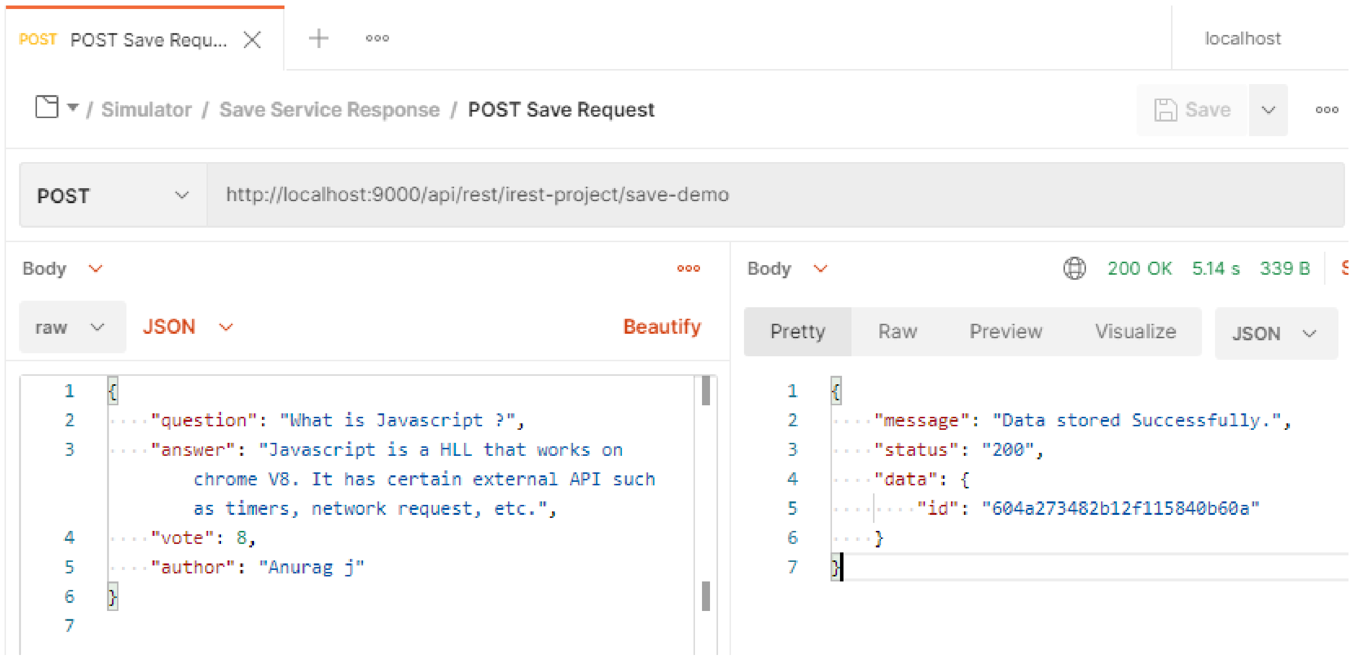
POST http://localhost:9000/api/rest/test-project-22334/endpoint-test

Status: 200 OK Time: 2.09 s Size: 27.63 KB

```
{
    "y": 120,
    "z": {},
    "qa": {
        "as": 1,
        "bf": 2
    },
    "abc": [],
    "kbc": [
        9,
        8,
        7,
        6,
        5
    ],
    "w": [
        {
            "k": [],
            "q": 123,
            "p": {},
            "r": [
                1,
                2,
                4
```

Annexure 3.

SIMULATE A SAVE POST API


POST POST Save Requ...
localhost
Simulator / Save Service Response / POST Save Request
Save
Body
raw JSON Beautify
Body
200 OK 5.14 s 339 B
Pretty Raw Preview Visualize JSON

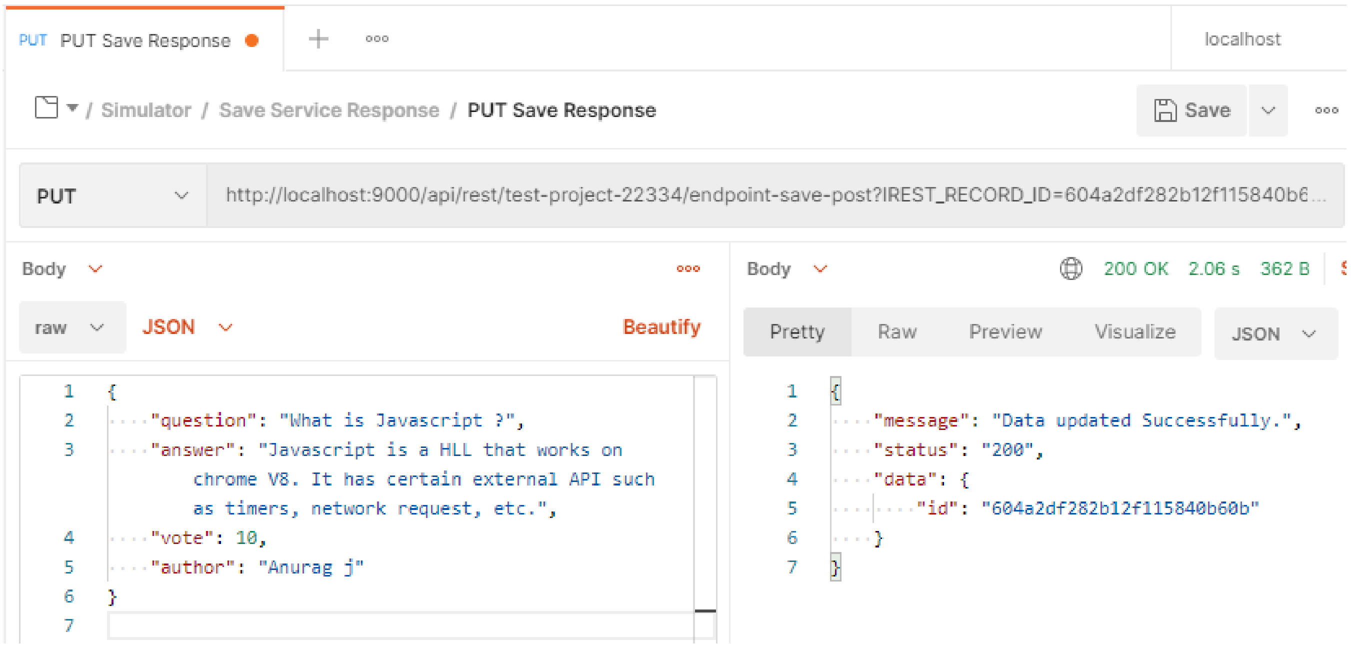
POST http://localhost:9000/api/rest/irest-project/save-demo

```
{
    "question": "What is Javascript ?",
    "answer": "Javascript is a HLL that works on
        chrome V8. It has certain external API such
        as timers, network request, etc.",
    "vote": 8,
    "author": "Anurag j"
}
```

```
{
    "message": "Data stored Successfully.",
    "status": "200",
    "data": {
        "id": "604a273482b12f115840b60a"
    }
}
```

Annexure 4.

SIMULATE A SAVE PUT API



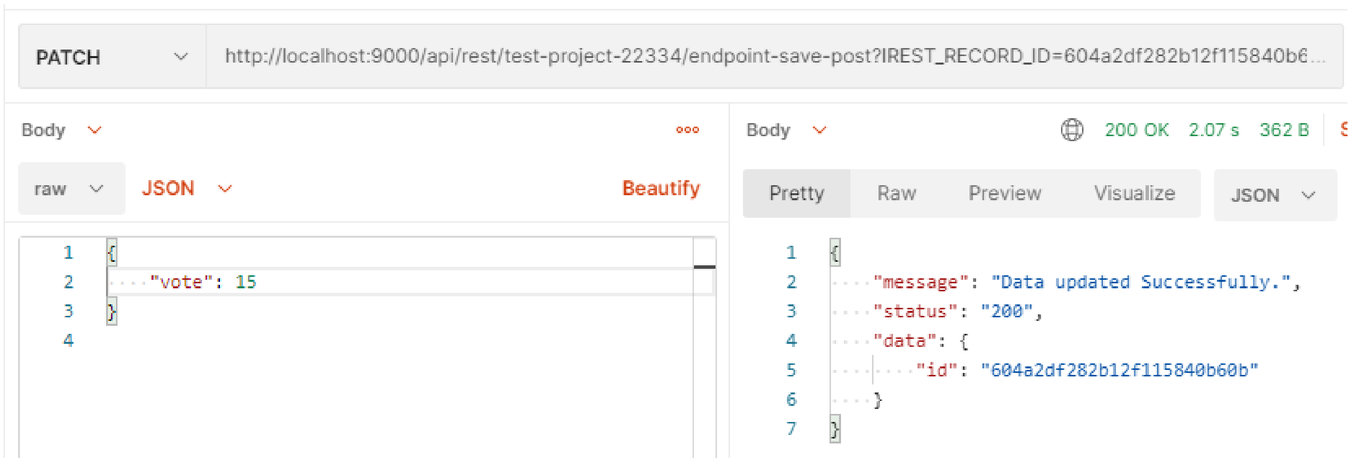
```
http://localhost:9000/api/rest/test-project-22334/endpoint-save-post?IREST_RECORD_ID=604a2df282b12f115840b6...
```

```
{
    "question": "What is Javascript ?",
    "answer": "Javascript is a HLL that works on
        chrome V8. It has certain external API such
        as timers, network request, etc.",
    "vote": 10,
    "author": "Anurag j"
}
```

```
{
    "message": "Data updated Successfully.",
    "status": "200",
    "data": {
        "id": "604a2df282b12f115840b60b"
    }
}
```

Annexure 5.

SIMULATE A SAVE PATCH API



PATCH http://localhost:9000/api/rest/test-project-22334/endpoint-save-post?IREST_RECORD_ID=604a2df282b12f115840b6…

Body raw JSON Beautify

```
{
    "vote": 15
}
```

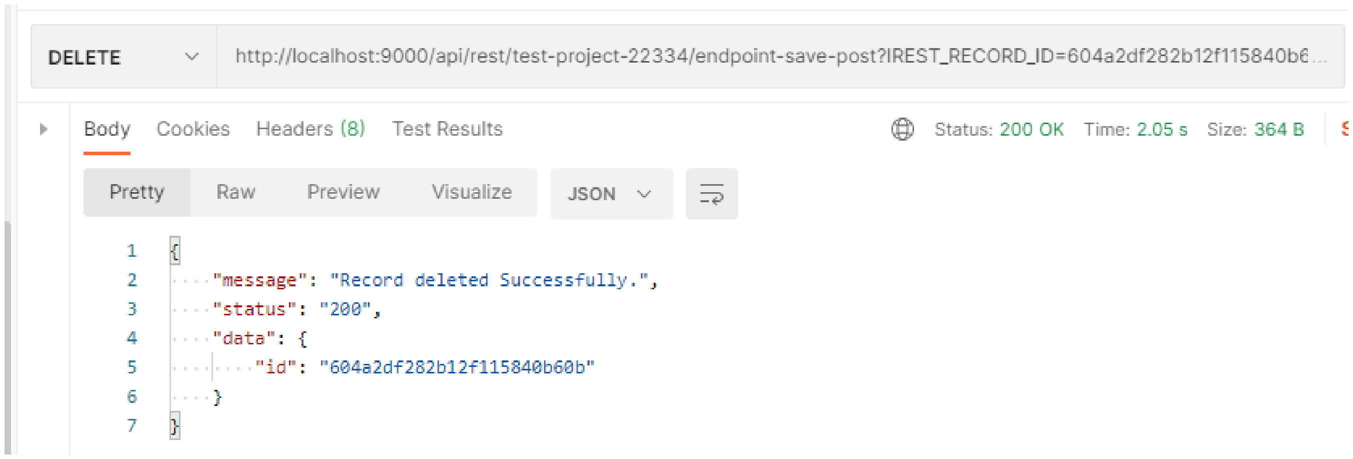
Body 200 OK 2.07 s 362 B

Pretty Raw Preview Visualize JSON

```
{
    "message": "Data updated Successfully.",
    "status": "200",
    "data": {
        "id": "604a2df282b12f115840b60b"
    }
}
```

Annexure 6.

SIMULATE A SAVE DELETE API



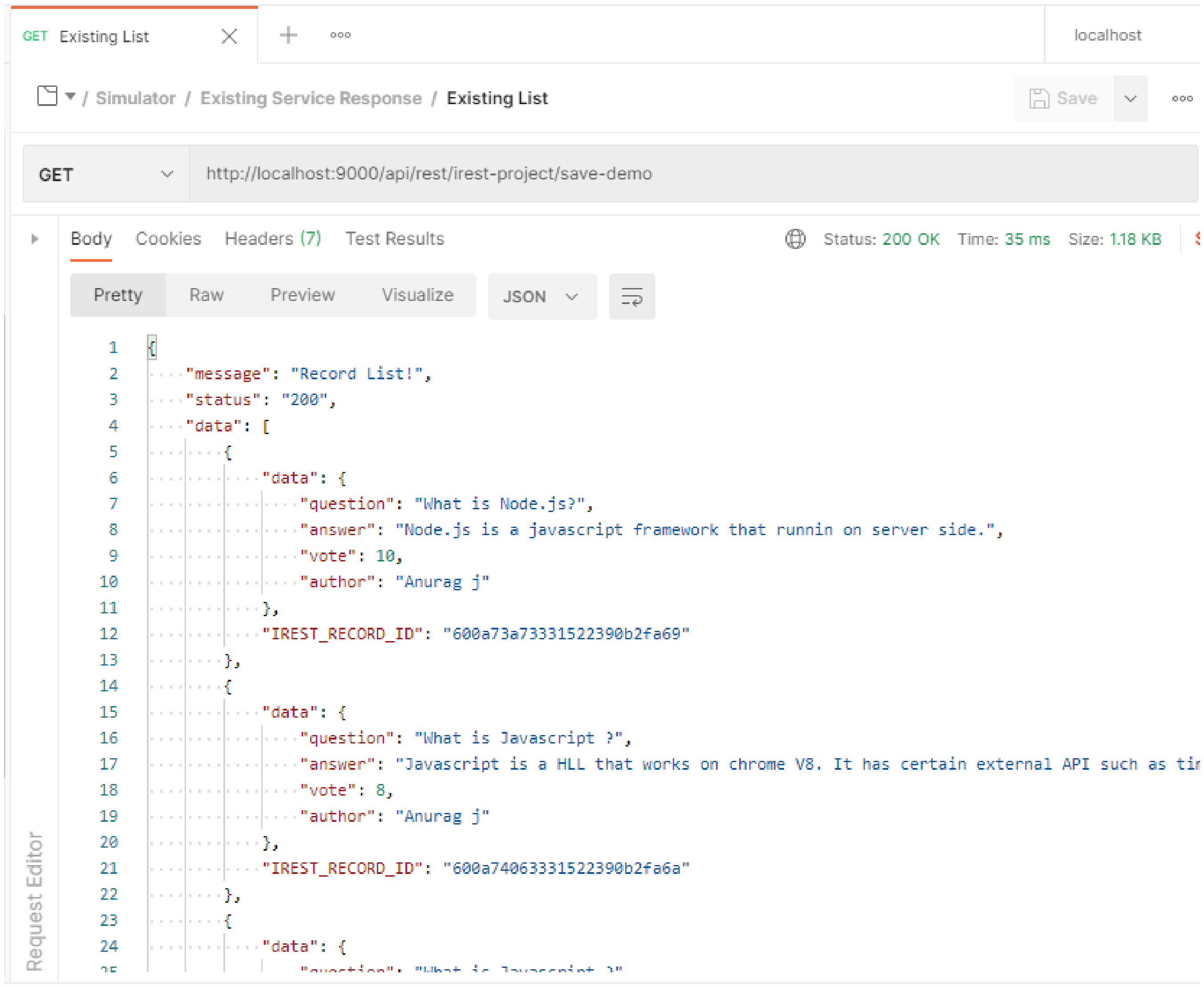
http://localhost:9000/api/rest/test-project-22334/endpoint-save-post?IREST_RECORD_ID=604a2df282b12f115840b6...

Status: 200 OK Time: 2.05 s Size: 364 B

```
{
    "message": "Record deleted Successfully.",
    "status": "200",
    "data": {
        "id": "604a2df282b12f115840b60b"
    }
}
```

Annexure 7.

SIMULATE A EXISTING API



GET http://localhost:9000/api/rest/irest-project/save-demo

Status: 200 OK Time: 35 ms Size: 1.18 KB

```
{
    "message": "Record List!",
    "status": "200",
    "data": [
        {
            "data": {
                "question": "What is Node.js?",
                "answer": "Node.js is a javascript framework that runnin on server side.",
                "vote": 10,
                "author": "Anurag j"
            },
            "IREST_RECORD_ID": "600a73a73331522390b2fa69"
        },
        {
            "data": {
                "question": "What is Javascript ?",
                "answer": "Javascript is a HLL that works on chrome V8. It has certain external API such as ti
                "vote": 8,
                "author": "Anurag j"
            },
            "IREST_RECORD_ID": "600a74063331522390b2fa6a"
        },
        {
            "data": {
```