Troubleshoot External API Calls or Service Requests from Designer

6/10/2026

Contents

Learn what happens when a Designer application makes an HTTP REST call using the HTTP REST block and how to troubleshoot issues with external API calls or service requests from Designer.

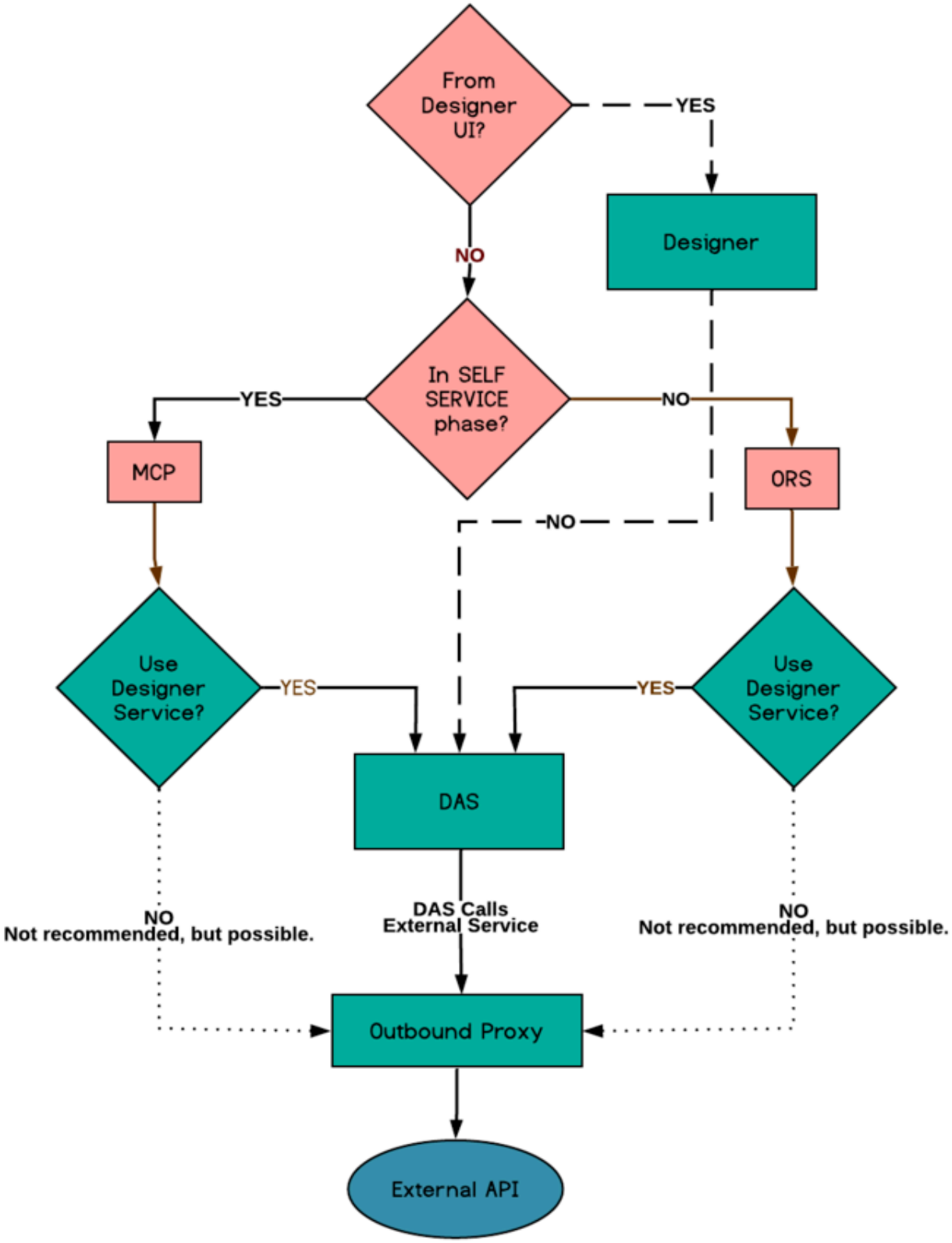
Overview

Applications created in Designer use the HTTP REST block to invoke target external APIs that a customer or any third party provides. Various things can go wrong while these APIs are invoked and in most cases the issue turns out to be unrelated to Designer. This article provides step-by-step directions to troubleshoot external API calls or service requests from Designer. It helps you identify the problem area and the root cause. More importantly, you are able to proceed in the right direction to resolve any API or connectivity issues. After following the troubleshooting steps outlined here, if the issue still persists, open a ticket with Genesys Customer Care and attach all the information described in this article.

Which components are involved?

There are various possible paths a request can take to hit an external API. The following flowchart illustrates the possible paths and the different components generally involved in such a request to an external API.

> **Important**
>
> The proxy in the diagram below is optional and may not be used in some environments.

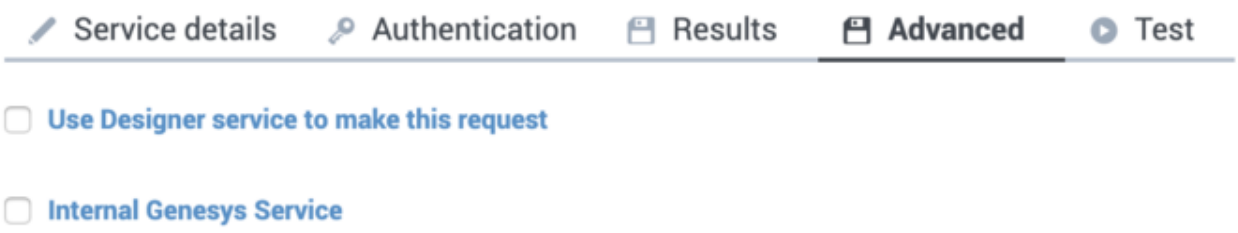

From Designer UI?
YES
Designer
NO
In SELF SERVICE phase?
YES
NO
MCP
ORS
NO
Use Designer Service?
YES
YES
Use Designer Service?
DAS
NO
Not recommended, but possible.
DAS Calls External Service
NO
Not recommended, but possible.
Outbound Proxy
External API

HTTP REST Troubleshooting

Step 1: Check the external service flow path

There are three possible paths through which a service request can be sent:

- via DAS (Check the Use Designer service to make this request option in the UI)
- via MCP (Check the Internal Genesys Service option in the UI)
- via ORS (Check the Internal Genesys Service option in the UI)



In the `externalrequests` section of the SDR, you can check whether the request is sent using a Designer service or MCP/ORS through the `proxied` field. In the example below, the first external request is sent via either MCP or ORS (depending on which phase the block is located), and then the second request is sent via the Designer service (PHP).


```
"externalrequests": [
      {
        "blockid": "7",
        "fetchduration": 557,
        "fetchend": "2020-03-19T22:53:16.454Z",
        "fetchstart": "2020-03-19T22:53:15.897Z",
        "name": "HTTP REST to get Access token from abc",
        "requestoutcome": {
          "httpProxy": "",
          "method": "post",
          "proxied": false,
          "responseCode": 200,
          "success": true,
          "url": "https://test.abc.com/services/oauth2/token",
          "useDelegate": false,
          "useHttpProxy": false
        },
        "seq": 1,
        "timedout": false,
        "timeout": 100,
        "type": "httpfetch"
      },
      {
        "blockid": "17",
        "fetchduration": 261,
        "fetchend": "2020-03-19T22:53:18.674Z",
        "fetchstart": "2020-03-19T22:53:18.413Z",
        "name": "HTTP REST access Claim Service",
        "requestoutcome": {
          "method": "get",
```

```
        "proxied": true,
        "responseCode": "500",
        "success": false,
        "url": "",
        "useDelegate": true,
        "useHttpProxy": false
      },
      "seq": 2,
      "timedout": false,
      "timeout": 100,
      "type": "httpfetch"
    }
]
```

> **Important**
>
> If the request is not `proxied`, then sending the external request is the responsibility of ORS and MCP. This troubleshooting article does not apply to those cases.

> **Important**
>
> The Designer Testing UI always sends external requests via the Designer PHP service. If you have requests that are handled by ORS/MCP, the testing UI will not be able to provide insight into those processes.

Step 2: Check if a proxy is sent with the request

In the above SDR snippet, for each external request, look for the `useHttpProxy` field. This field signals whether a `httpProxy` is being sent along with the request or not. If the value is *false*, it means a `httpProxy` is not sent, and thus not used at all. If you see the `useHttpProxy` value to be *true*, perform the next step to check if `httpProxy` is set to the correct value; if `useHttpProxy` is *false* (but a proxy is needed), perform the next step to check if the setting is missing.

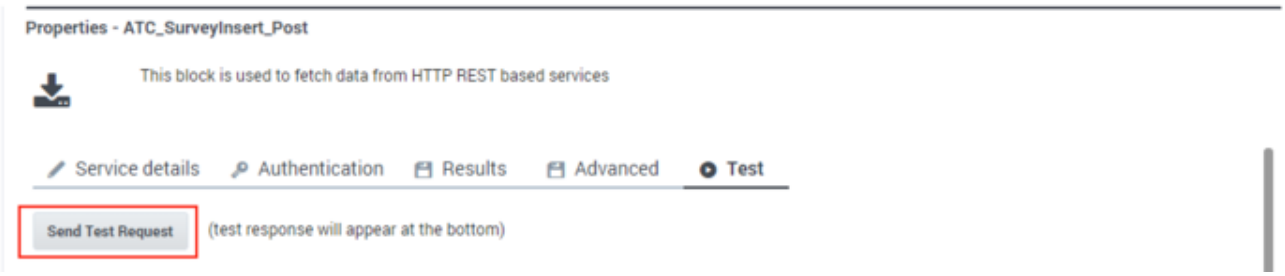
Step 3: Check httpProxy setting in Designer

External API URL(s) are added to the proxy server's allowlist for both Designer and DAS.

- If `httpProxy` is set correctly at this level, go back to **Step 2**.
- If `httpProxy` is missing at this level, enable proxy settings in Designer, to talk to external APIs.
- If `httpProxy` is set incorrectly, modify the **DesignerEnv** transaction list to reflect the correct proxy setting.

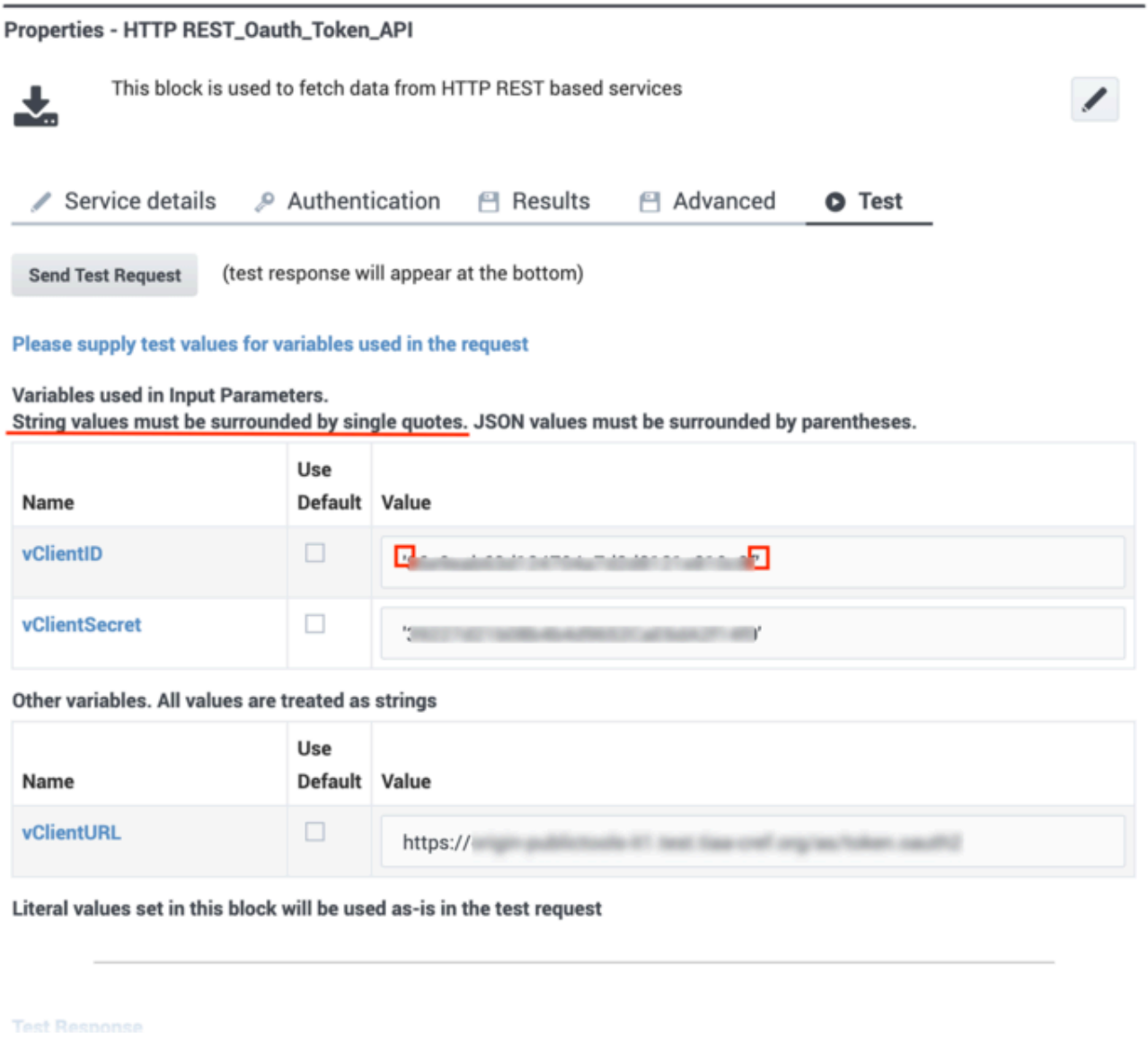
Step 4: Use the Designer UI to test the external service

The HTTP Rest block in Designer has a testing function, which you can use to test connectivity to the external service. Supply variables with proper values and click the **Send Test Request** button.

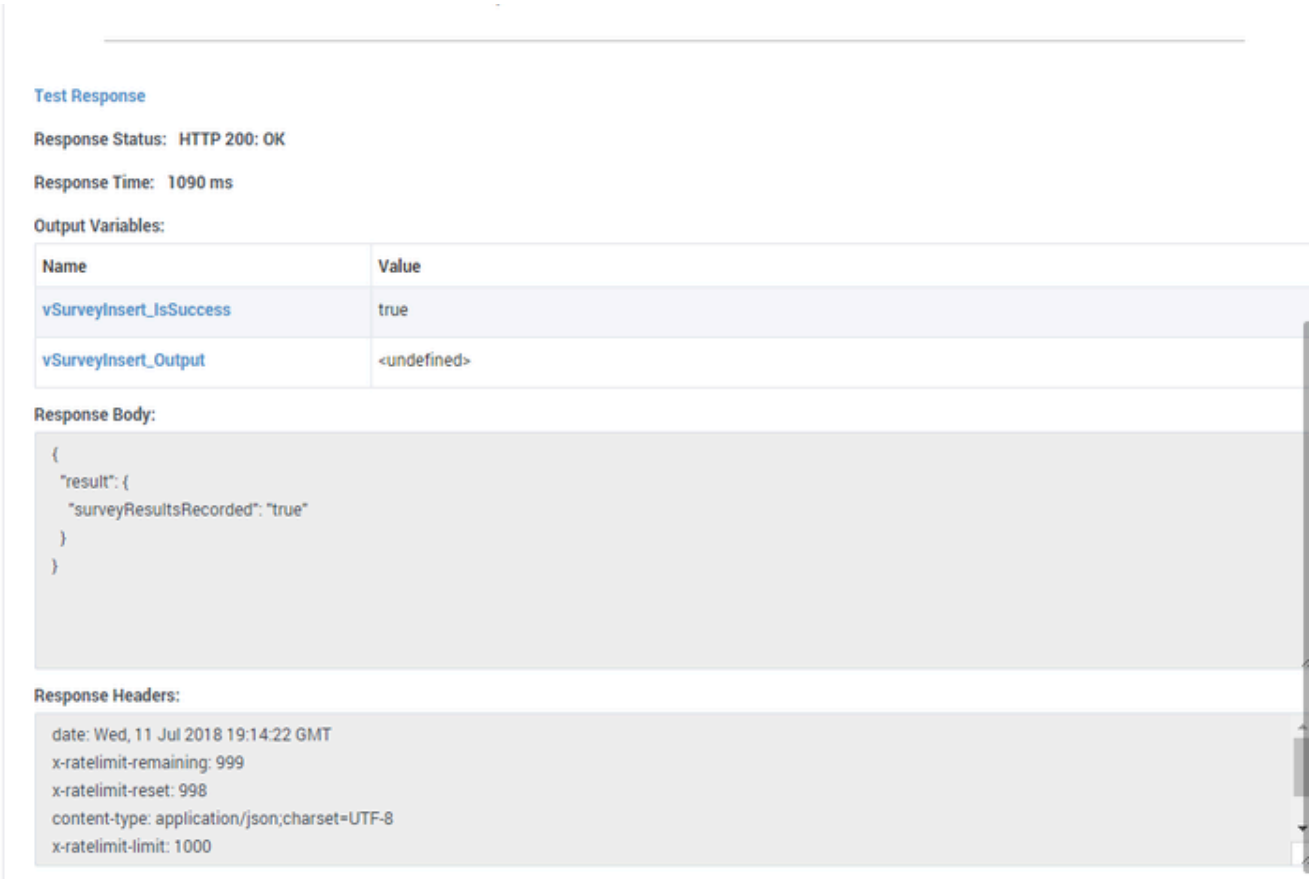


Important

If you are supplying values for variables used as input parameters, surround the string values with single quotes. JSON values must be surrounded by parentheses.



You must verify the following in the response:



1. The **Response Status** code must be **200** (if not, make sure that the external service is functioning. Considering the input, without involving `httpProxy` and using Postman or direct cURL).
2. **Response Time** must be within the range specified in the **Request Timeout** field in the block (if the request takes longer than the specified timeout value, you must adjust the timeout value accordingly.)
3. **Response Body** must be in the same format as the **Response Header**'s content-type (if there is any discrepancy, the external service must be revised in order to consume the output).
4. Output variables, if set, must be processed according to the response given (if there is any discrepancy, the application must be revised in order to consume the output).
5. If a proxy is used, verify that the correct httpProxy is used in the `x-genesys-proxy` property on the response headers (if there is any discrepancy, the `flowsettings` or `DesignerEnv` must be fixed).

If the above checks fail and you have followed all the previous steps, skip the following step and proceed to **Step 6**. If these checks pass, move back to **Step 3**.

Step 5: Verify the external service (with `httpProxy`) on application servers

You must test the external service URL based on the communication contract set by the customer. It is ideal to test these external services independently, either through cURL or a tool like Postman. But most of the times we can only test through the proxy set up in **Step 1**. The test must be done on the application servers that will eventually send the requests to an external service.

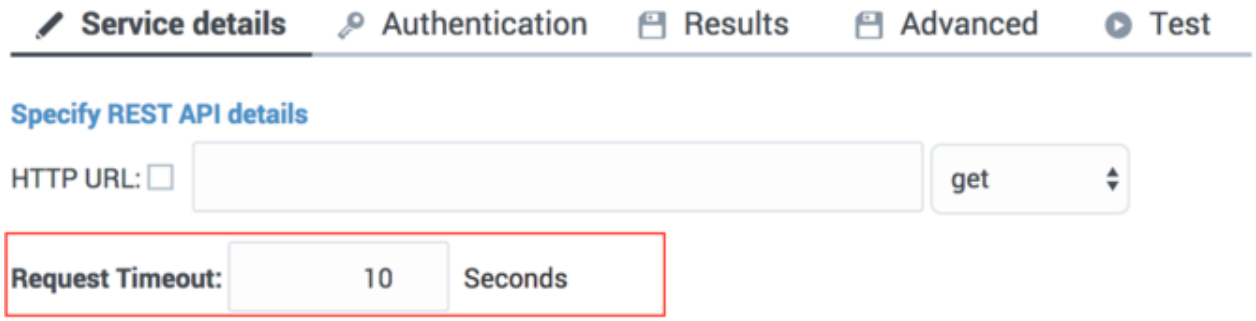
Here is an example for directly curling a customer service:

```
curl -x "http://:" -XPOST "https://" -d "user=abc&password=xyz" -k -H "Content-Type:
application/x-www-form-urlencoded"
```

During this step, it is important to keep track of the total time taken for a response to be received after the request has been sent. You can do this using the following cURL command:

```
curl -XGET [external URL] -w '\nTotal: %{time_total}\n'
```

The total time of the request must be less than the **Request Timeout** property set in the HTTP REST block; otherwise the request returns a timeout error.



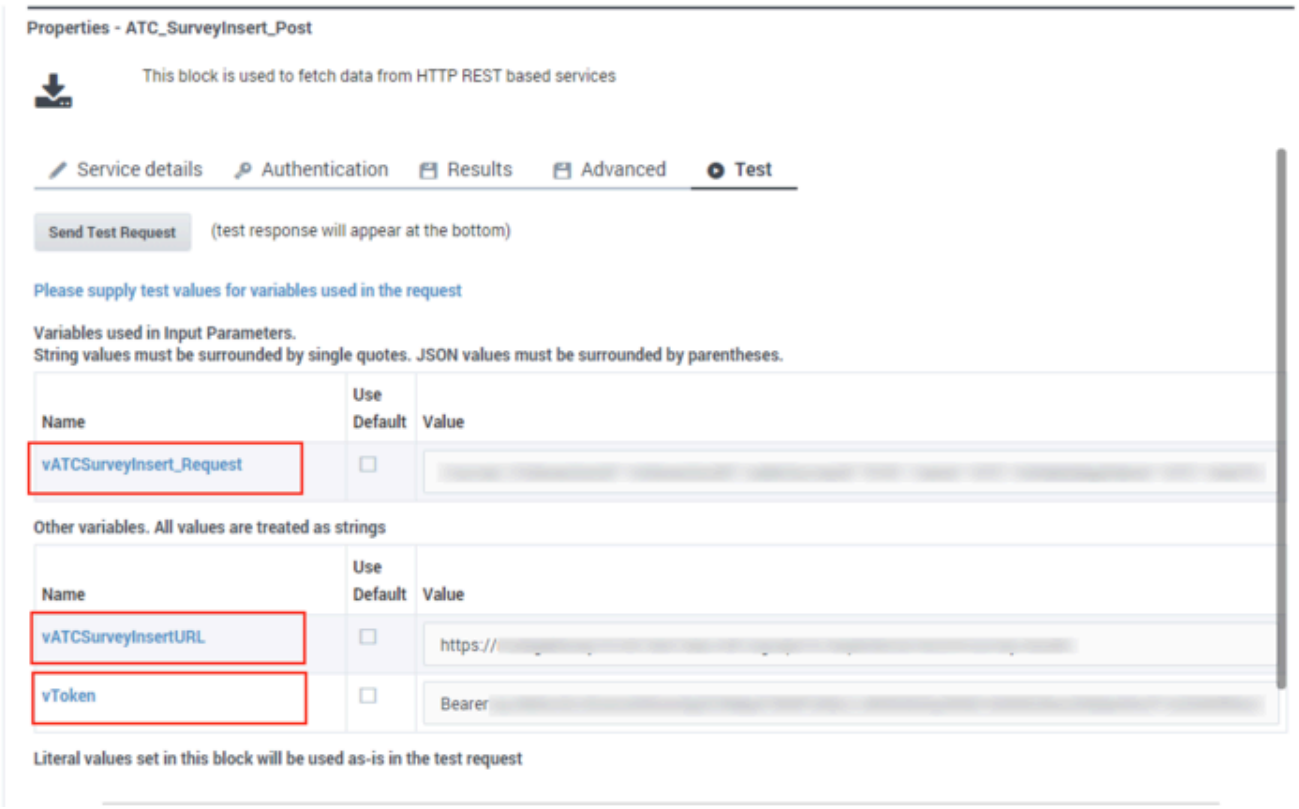
In this step,

- If the check fails on the external service itself, you must contact the external service provider to verify the service contract.
- If the external service works by itself but not through the proxy, you must ensure that it is set up properly.
- If the proxy connection does not work with some application servers, you must ensure that those app servers are configured properly.
- If the request takes longer than the timeout specified in the UI, you must adjust the specification so it waits long enough.

If these checks pass, go back to **Step 4**.

Step 6: Verify Request Payload/Parameter in SDR

If Analytics is enabled, you can locate the problematic SDR session (which contains the external service) by using Kibana. For example, in the screenshot below, the HTTP REST block sends the payload recorded by the `vATCSurveyInsert_Request` variable toward the URL recorded in `vATCSurveyInsertURL`. And the token value comes from `vToken` in the request header.



The variable values can be verified in the corresponding SDR's variables section:

```
"variables": {
      ......
      "vATCSurveyInsertURL": "",
      "vToken": "Bearer ...........",
      ......
    }
```

If this check fails, the application is not set up properly to generate the correct variable values to pass along to the external request. The application developer must verify the application flow and setup, specifically, how the JSON object is constructed and passed on to the external service.

If this check passes, go back to **Step 4**.

Step 7: cURL Designer PHP service via DAS

It is helpful to directly cURL the Designer PHP service that issues external requests to ensure that everything is working on the application server side.

Perform the following and record your results and go back to **Step 6**:

1. Send a request using key-value pairs:

   ```
   curl -XPOST "//systemservice/httprest/service.php" -d
   {
   ```

```
  "url": "https://login.abc.com/services/oauth2/token",
  "method": "post",
  "inputParams": [
    {
      "name": "grant_type",
      "value": "password"
    },
    {
      "name": "client_id",
      "value": "........"
    },
    {
      "name": "client_secret",
      "value": "............."
    },
    {
      "name": "username",
      "value": "abc@abc.com"
    },
    {
      "name": "password",
      "value": "abc1"
    }
  ],
  "timeout": 10,
  "headers": [
  ],
  "isgenesysinternal": false
}
```

2. Send a request using a JSON payload:

```
curl -XPOST http://localhost:3000/httprest -H "Content-Type:application/json" -d '
{
       "httpProxy": ":",
       "url": "",
       "method": "post",
       "json": {
              "user": ""
       },
       "headers": [{"name":"something", "value":"else"}]
}'
```

3. Send a request using custom headers:

```
curl -XPOST "http://localhost/.........../systemservice/httprest/service.php" -d

 {
  "url": "",
  "method": "get",
  "inputParams": [],
  "timeout": 10,
  "headers": [
    {
      "name": "Authorization",
      "value": "Bearer ......"
    }
  ],
  "isgenesysinternal": false
}
```

Step 8: Report issue to Genesys Customer Care

Here is a list of questions to consider before reporting external request issues to Genesys Customer Care for further investigation:

Question	Things to consider
Was this external request working before in this particular application?	If yes, what changed (Designer upgrade, application republish, external service update, application server config change)?
Is the same external request known to be working among other applications of the tenant?	If so, which ones?
Are other external requests working, or do all of them resort to failure?	Yes or No?

Also, keep the following ready to provide to the customer care representative.

- Application/module information.
- Testable input parameters and payload values.
- Test results from steps 1-5 of this documentation (including cURL requests used in testing, and responses received).
- SDR which shows external request failure in a particular session.
- ORS logs.
- nginx_access.log and php_error logs (DAS generates both these types of logs).

> **Important**
>
> Custom services are NOT supported by Designer in its premise and private edition versions.