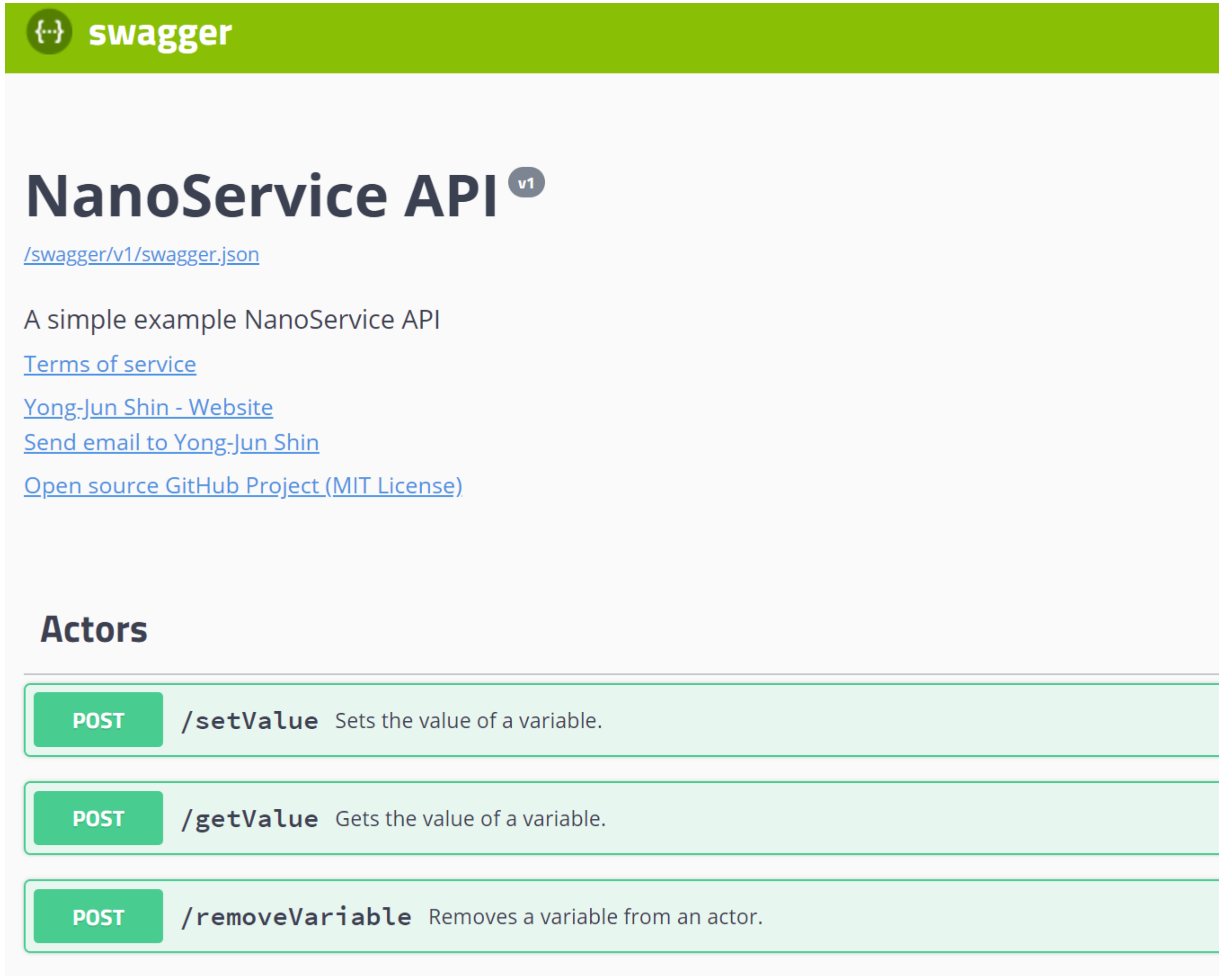
Nanoservice-enabled event-driven clinical decision support REST APIs

Hyponatremia Diagnosis Demo

For more information about Nanoservice, please visit https://csml.uconn.edu/

Step 1. Go to http://csmlab7.uconn.edu/swagger/



NanoService API v1

/swagger/v1/swagger.json

A simple example NanoService API

Terms of service

Yong-Jun Shin - Website

Send email to Yong-Jun Shin

Open source GitHub Project (MIT License)

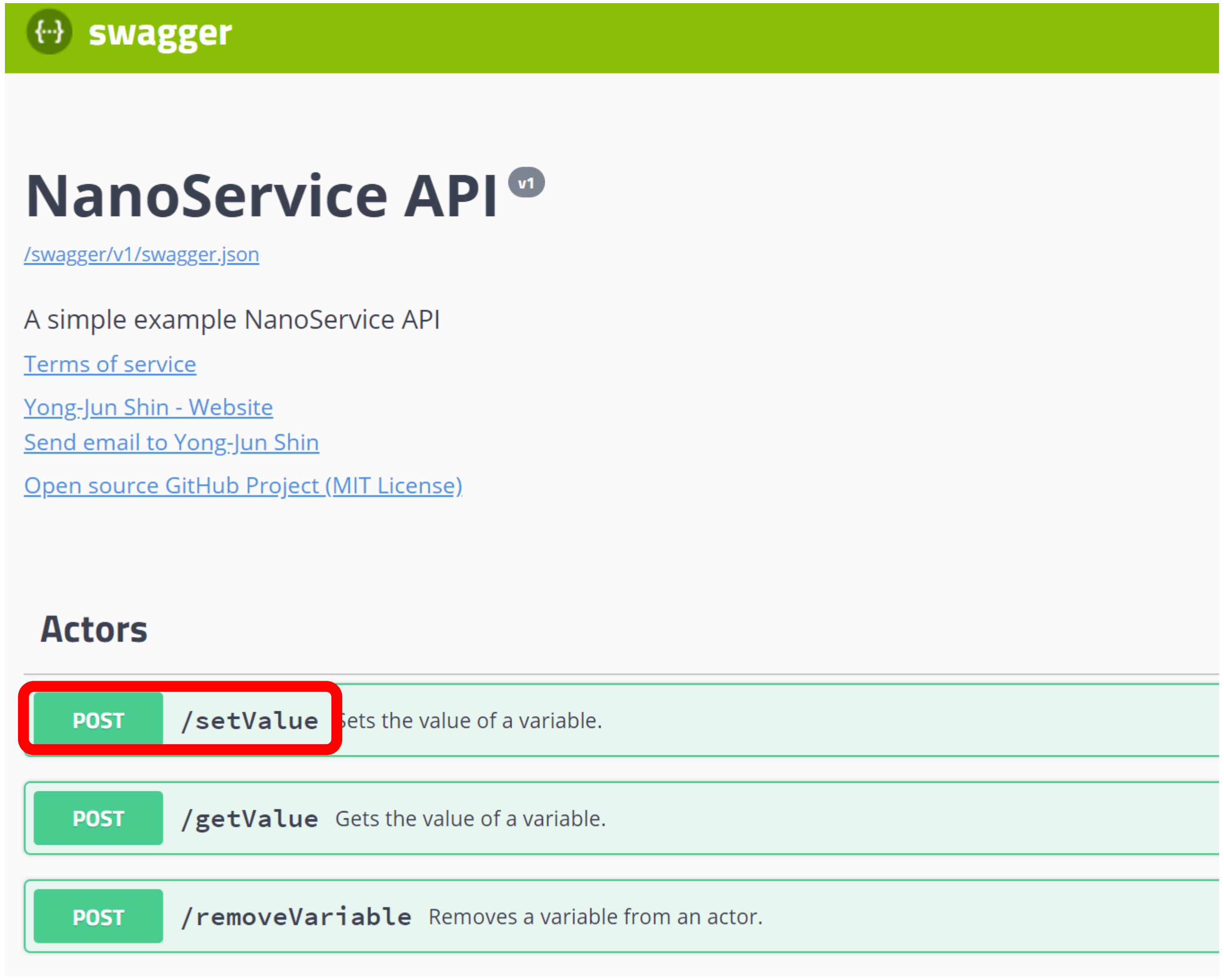
Actors

POST /setValue Sets the value of a variable.

POST /getValue Gets the value of a variable.

POST /removeVariable Removes a variable from an actor.

Step 2. [setValue] Click “[POST] /setValue”



NanoService API v1

/swagger/v1/swagger.json

A simple example NanoService API

Terms of service

Yong-Jun Shin - Website

Send email to Yong-Jun Shin

Open source GitHub Project (MIT License)

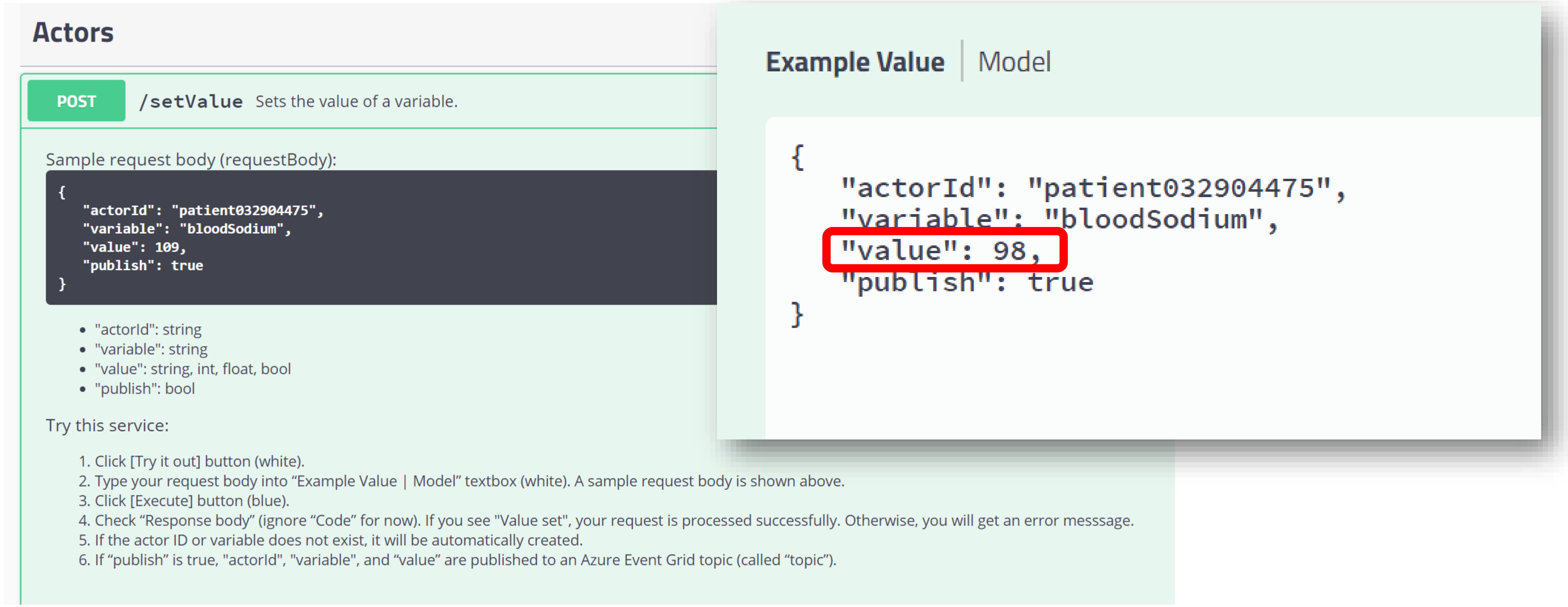
Actors

POST /setValue Sets the value of a variable.

POST /getValue Gets the value of a variable.

POST /removeVariable Removes a variable from an actor.

Step 3. [setValue] Set the value of the variable “bloodSodium” to 98 by following the instruction.



Actors

/setValue Sets the value of a variable.

Sample request body (requestBody):

```
{
   "actorId": "patient032904475",
   "variable": "bloodSodium",
   "value": 109,
   "publish": true
}
```

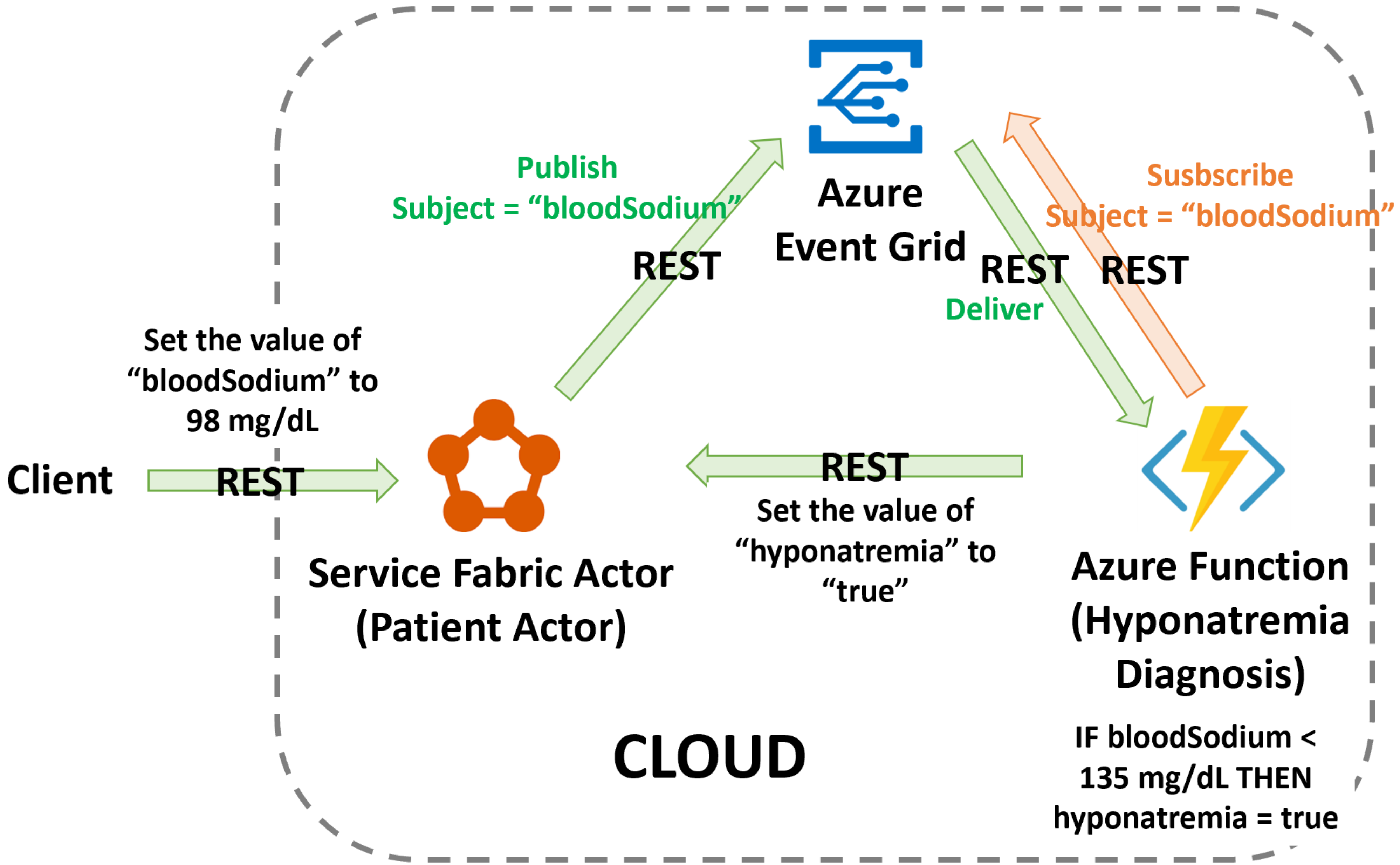
- "actorId": string
- "variable": string
- "value": string, int, float, bool
- "publish": bool

Try this service:

1. Click [Try it out] button (white).
2. Type your request body into “Example Value | Model” textbox (white). A sample request body is shown above.
3. Click [Execute] button (blue).
4. Check “Response body” (ignore “Code” for now). If you see "Value set", your request is processed successfully. Otherwise, you will get an error messsage.
5. If the actor ID or variable does not exist, it will be automatically created.
6. If “publish” is true, "actorId", "variable", and “value” are published to an Azure Event Grid topic (called “topic”).

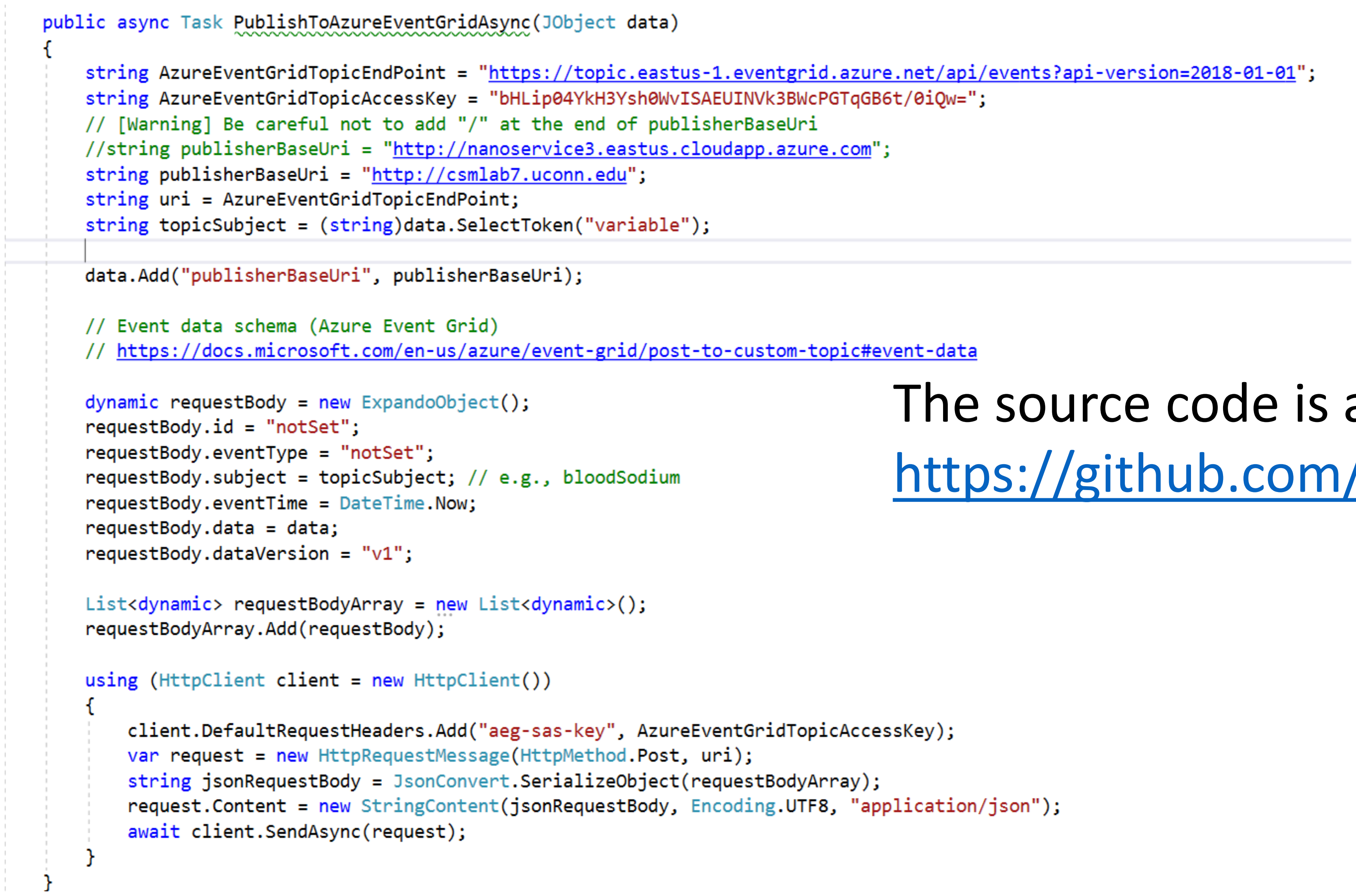
```
{
   "actorId": "patient032904475",
   "variable": "bloodSodium",
   "value": 98,
   "publish": true
}
```

- If the actor ID or variable does not exist, it will be automatically created.
- If “publish” is true, "actorId", "variable", and “value” are published to an Azure Event Grid topic (called “topic”).


Publish
Subject = “bloodSodium”
REST
Azure
Event Grid
REST
Deliver
REST
Susbscribe
Subject = “bloodSodium”
Set the value of
“bloodSodium” to
98 mg/dL
Client
REST
Service Fabric Actor
(Patient Actor)
REST
Set the value of
“hyponatremia” to
“true”
Azure Function
(Hyponatremia
Diagnosis)
IF bloodSodium <
135 mg/dL THEN
hyponatremia = true
CLOUD

The code for publishing an event to **Azure Event Grid**

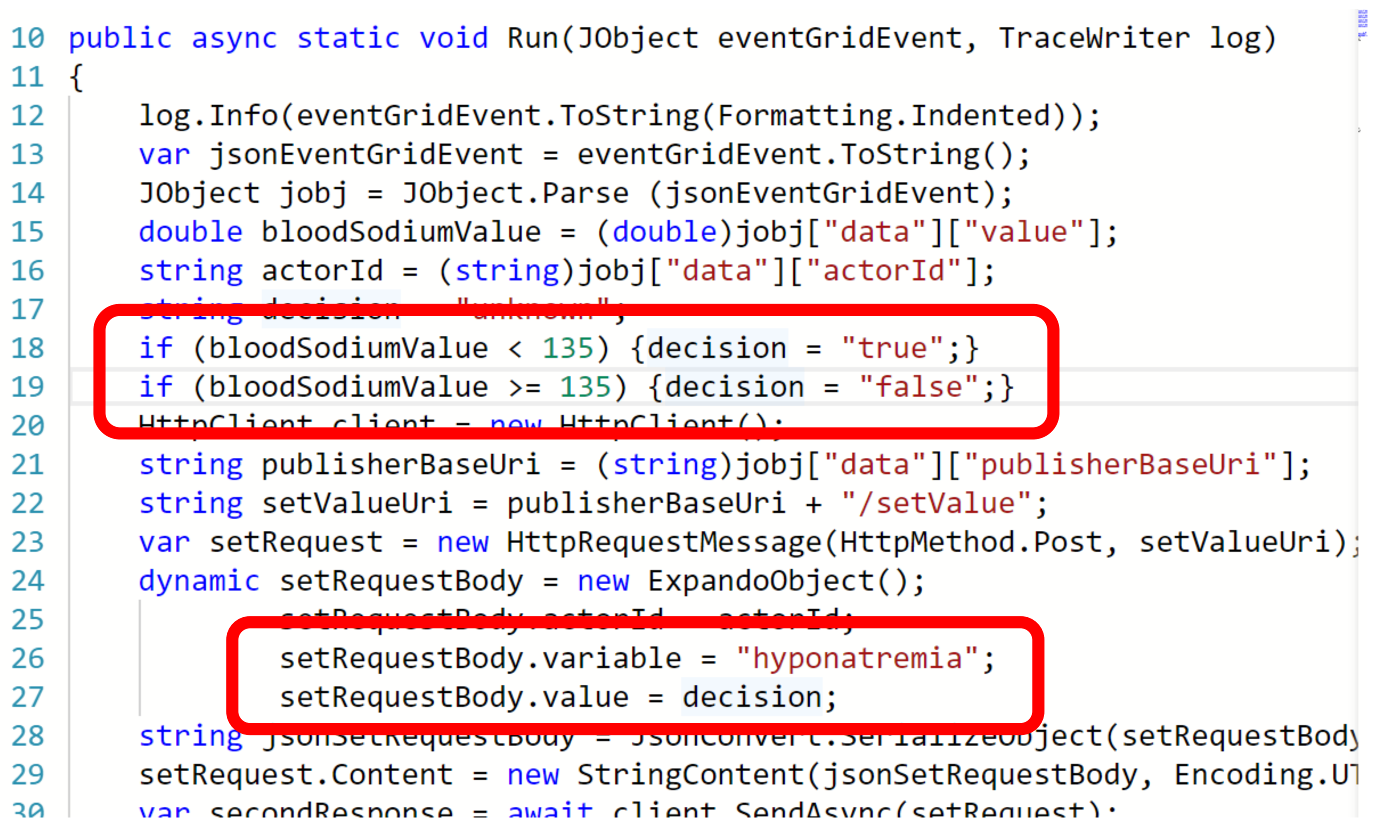
https://azure.microsoft.com/en-us/services/event-grid/



```
public async Task PublishToAzureEventGridAsync(JObject data)
{
    string AzureEventGridTopicEndPoint = "https://topic.eastus-1.eventgrid.azure.net/api/events?api-version=2018-01-01";
    string AzureEventGridTopicAccessKey = "bHLip04YkH3Ysh0WvISAEUINVk3BWcPGTqGB6t/0iQw=";
    // [Warning] Be careful not to add "/" at the end of publisherBaseUri
    //string publisherBaseUri = "http://nanoservice3.eastus.cloudapp.azure.com";
    string publisherBaseUri = "http://csmlab7.uconn.edu";
    string uri = AzureEventGridTopicEndPoint;
    string topicSubject = (string)data.SelectToken("variable");

    data.Add("publisherBaseUri", publisherBaseUri);

    // Event data schema (Azure Event Grid)
    // https://docs.microsoft.com/en-us/azure/event-grid/post-to-custom-topic#event-data

    dynamic requestBody = new ExpandoObject();
    requestBody.id = "notSet";
    requestBody.eventType = "notSet";
    requestBody.subject = topicSubject; // e.g., bloodSodium
    requestBody.eventTime = DateTime.Now;
    requestBody.data = data;
    requestBody.dataVersion = "v1";

    List<dynamic> requestBodyArray = new List<dynamic>();
    requestBodyArray.Add(requestBody);

    using (HttpClient client = new HttpClient())
    {
        client.DefaultRequestHeaders.Add("aeg-sas-key", AzureEventGridTopicAccessKey);
        var request = new HttpRequestMessage(HttpMethod.Post, uri);
        string jsonRequestBody = JsonConvert.SerializeObject(requestBodyArray);
        request.Content = new StringContent(jsonRequestBody, Encoding.UTF8, "application/json");
        await client.SendAsync(request);
    }
}
```

The source code is available at:
https://github.com/yshin1209/Nanoservice

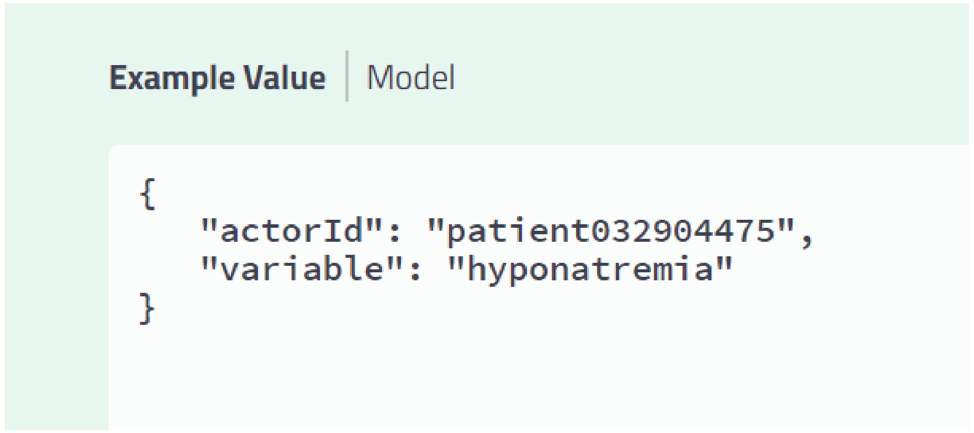
Azure Function https://azure.microsoft.com/en-us/services/functions/



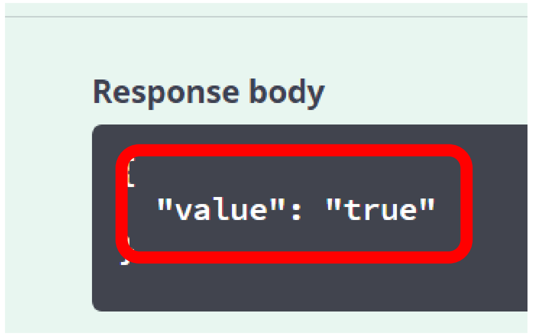
```
public async static void Run(JObject eventGridEvent, TraceWriter log)
{
    log.Info(eventGridEvent.ToString(Formatting.Indented));
    var jsonEventGridEvent = eventGridEvent.ToString();
    JObject jobj = JObject.Parse (jsonEventGridEvent);
    double bloodSodiumValue = (double)jobj["data"]["value"];
    string actorId = (string)jobj["data"]["actorId"];
    string decision = "unknown";
    if (bloodSodiumValue < 135) {decision = "true";}
    if (bloodSodiumValue >= 135) {decision = "false";}
    HttpClient client = new HttpClient();
    string publisherBaseUri = (string)jobj["data"]["publisherBaseUri"];
    string setValueUri = publisherBaseUri + "/setValue";
    var setRequest = new HttpRequestMessage(HttpMethod.Post, setValueUri);
    dynamic setRequestBody = new ExpandoObject();
            setRequestBody.actorId = actorId;
            setRequestBody.variable = "hyponatremia";
            setRequestBody.value = decision;
    string jsonSetRequestBody = JsonConvert.SerializeObject(setRequestBody
    setRequest.Content = new StringContent(jsonSetRequestBody, Encoding.U
    var secondResponse = await client.SendAsync(setRequest);
```

The Hyponatremia Diagnosis Function 1) decides whether there is hyponatremia or not, 2) adds a new variable “hyponatremia” to the patient actor in case the variable is not yet added and sets the value to true (hyponatremia) or false (not hyponatremia).

Step 4. [getValue] Get the value of the variable “hyponatremia” which is now “true” (because 98 < 135)

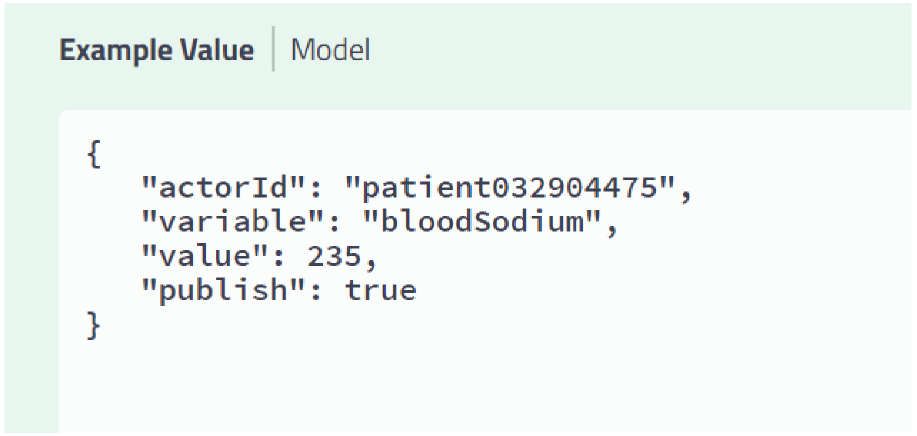

Example Value | Model

```
{
    "actorId": "patient032904475",
    "variable": "hyponatremia"
}
```


Response body

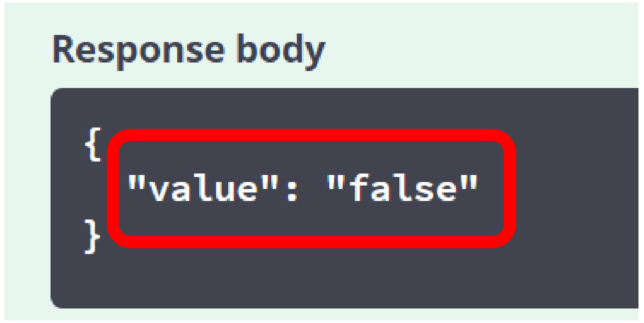
```
{
  "value": "true"
}
```

Step 5. [setValue] Set the value of the variable “bloodSodium” to 235.



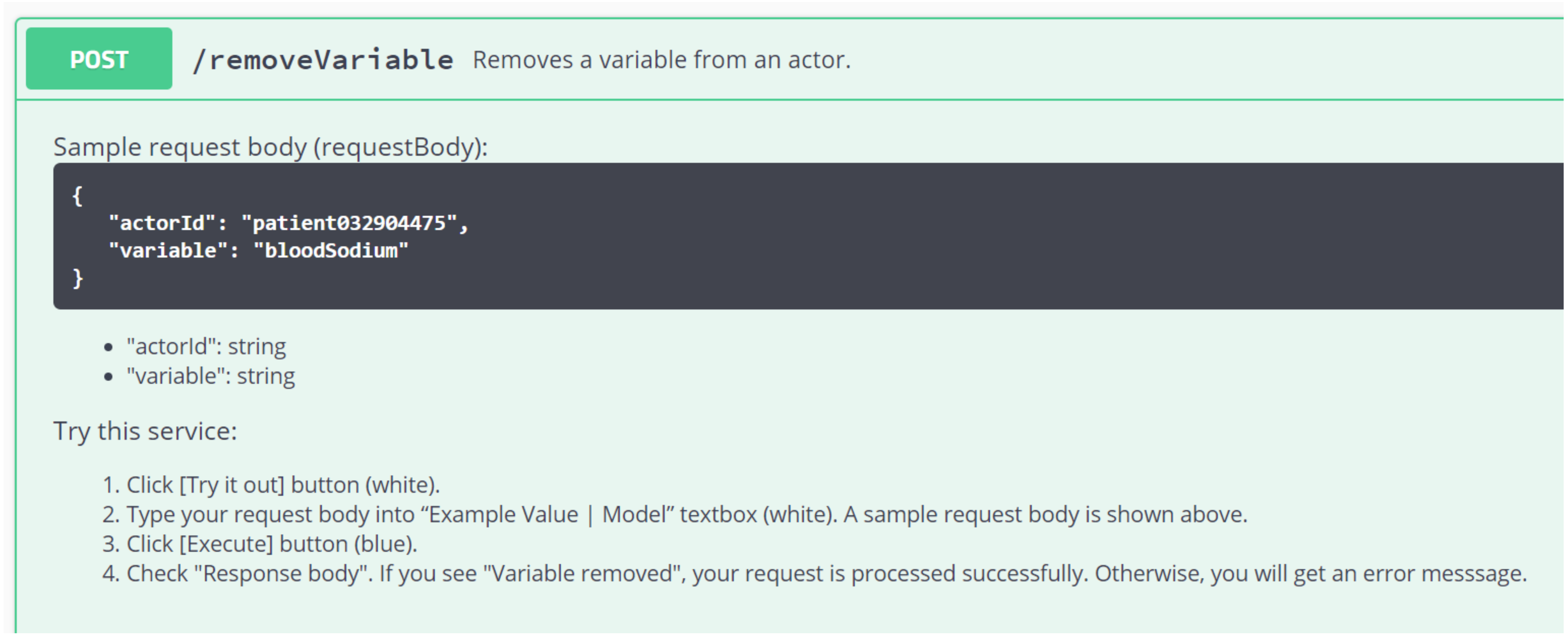
```
{
   "actorId": "patient032904475",
   "variable": "bloodSodium",
   "value": 235,
   "publish": true
}
```

Step 6. [getValue] Get the value of the variable “hyponatremia” which is now “false” (because 235 > 135)



```
{
  "value": "false"
}
```

Step 7. [removeVariable] Remove the variable “bloodSodium”.



POST `/removeVariable` Removes a variable from an actor.

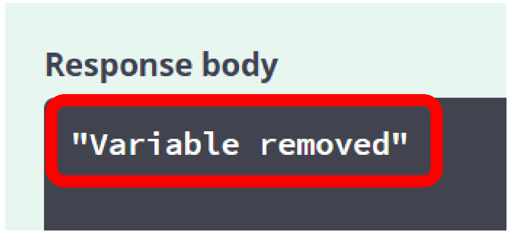
Sample request body (requestBody):

```
{
    "actorId": "patient032904475",
    "variable": "bloodSodium"
}
```

- "actorId": string
- "variable": string

Try this service:

1. Click [Try it out] button (white).
2. Type your request body into “Example Value | Model” textbox (white). A sample request body is shown above.
3. Click [Execute] button (blue).
4. Check "Response body". If you see "Variable removed", your request is processed successfully. Otherwise, you will get an error messsage.



Response body

```
"Variable removed"
```

For more information about Nanoservice, please visit https://csml.uconn.edu/