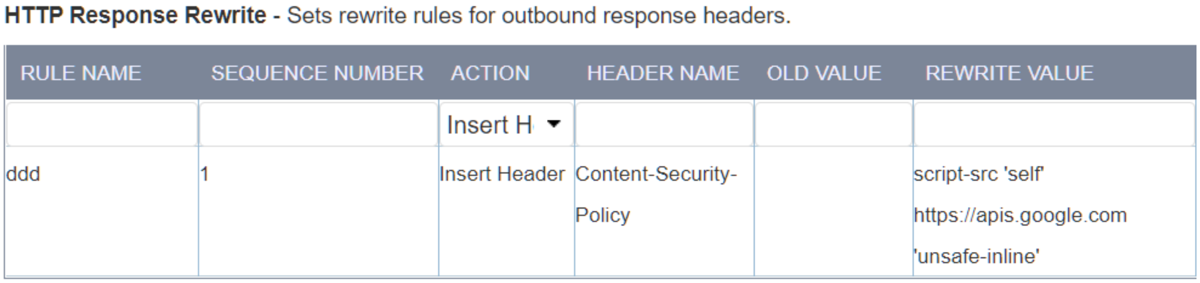
WAF Petstore API Lab Guide

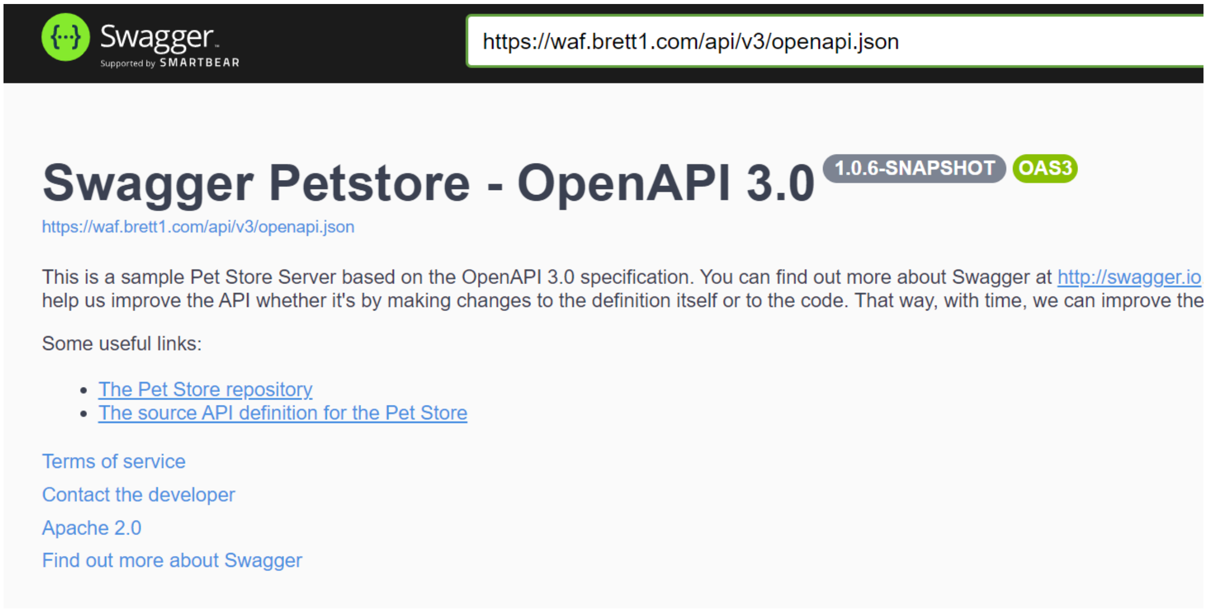
1. Login to your WAF, then create a service listening on 443 with a server named badstore-origin.cudathon.com listening on 8080
2. WAF->Websites -> Website Translations add the following



HTTP Response Rewrite - Sets rewrite rules for outbound response headers.

RULE NAME	SEQUENCE NUMBER	ACTION	HEADER NAME	OLD VALUE	REWRITE VALUE
		Insert H ▾			
ddd	1	Insert Header	Content-Security-Policy		script-src 'self' https://apis.google.com 'unsafe-inline'

3. Download the Petstore API Swagger file from https://petstore.swagger.io/v2/swagger.json to your computer
4. WAF->Websites->JSON Security-JSON Security->Import API Spec and choose the swagger.json you downloaded
5. Browse to the Service and verify you see the web page



https://waf.brett1.com/api/v3/openapi.json

Swagger Petstore - OpenAPI 3.0 1.0.6-SNAPSHOT OAS3

https://waf.brett1.com/api/v3/openapi.json

This is a sample Pet Store Server based on the OpenAPI 3.0 specification. You can find out more about Swagger at http://swagger.io
help us improve the API whether it's by making changes to the definition itself or to the code. That way, with time, we can improve the

Some useful links:

- The Pet Store repository
- The source API definition for the Pet Store

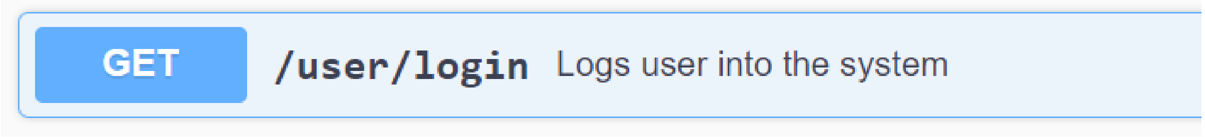
Terms of service

Contact the developer

Apache 2.0

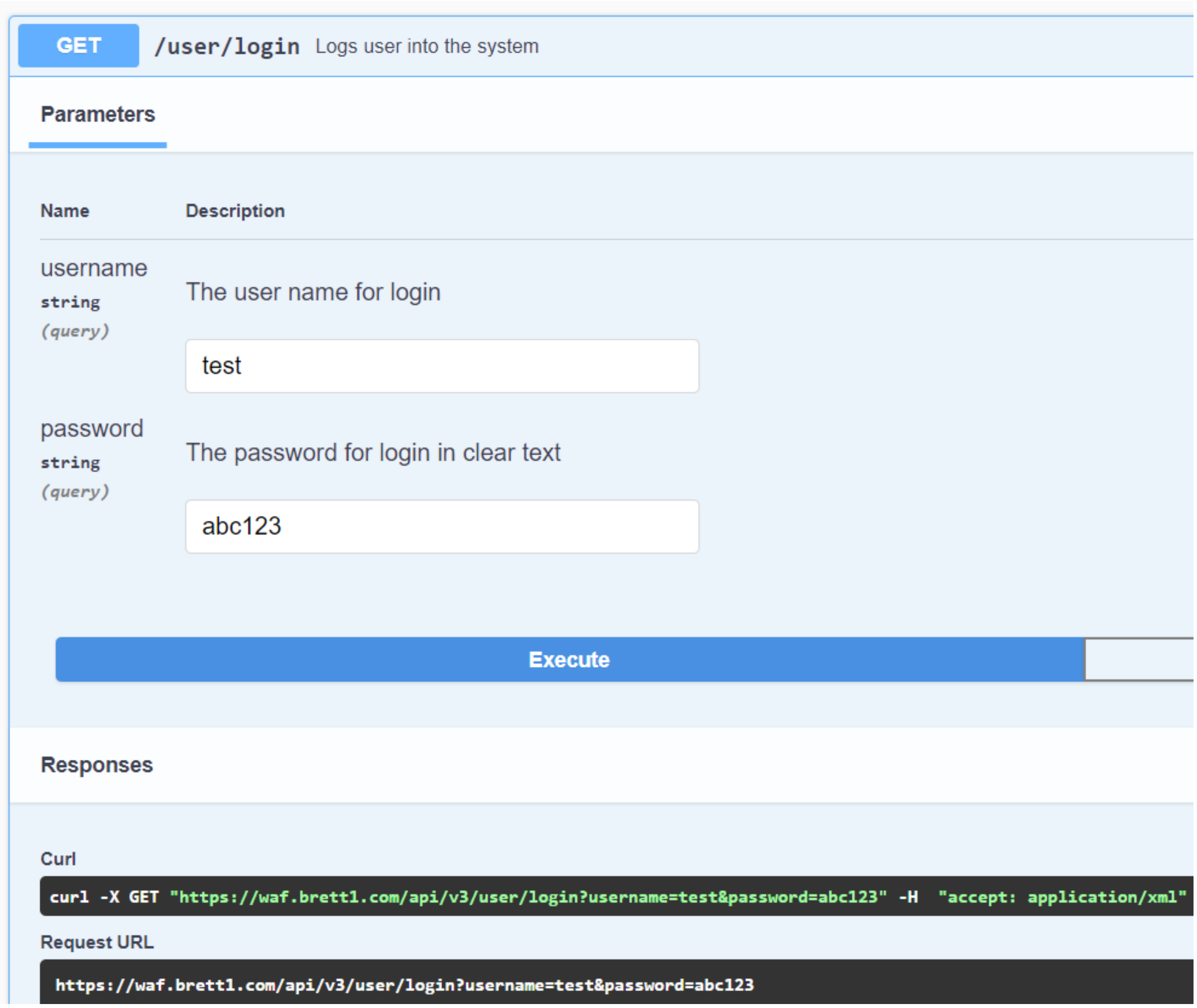
Find out more about Swagger

6. Scroll down and click on GET user/login



GET /user/login Logs user into the system

7. Click “Try it Out” and fill in username and passsword as test/abc123 then click execute



Name	Description
username string *(query)*	The user name for login test
password string *(query)*	The password for login in clear text abc123

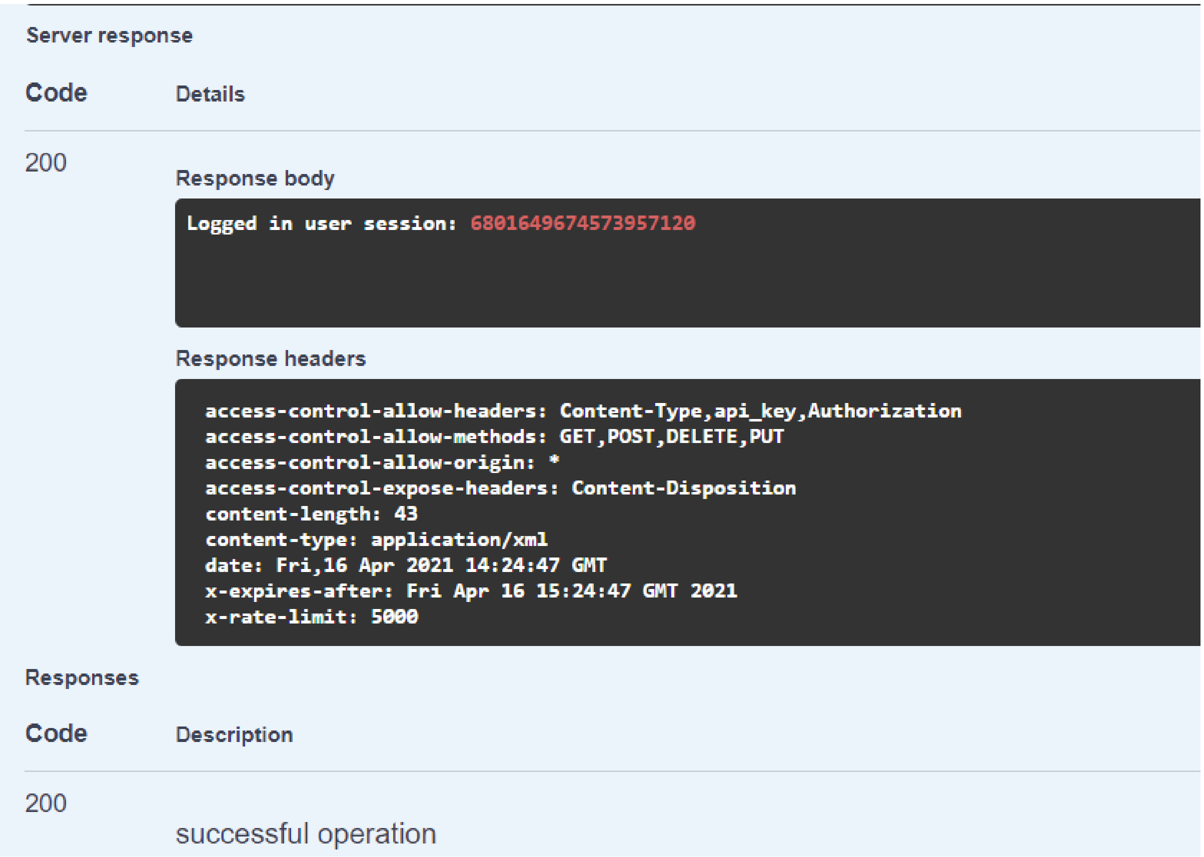
Curl

```
curl -X GET "https://waf.brett1.com/api/v3/user/login?username=test&password=abc123" -H  "accept: application/xml"
```

Request URL

```
https://waf.brett1.com/api/v3/user/login?username=test&password=abc123
```

8. Verify you see the following response (session id will be different)



Server response

Code	Details
200	Response body

```
Logged in user session: 6801649674573957120
```

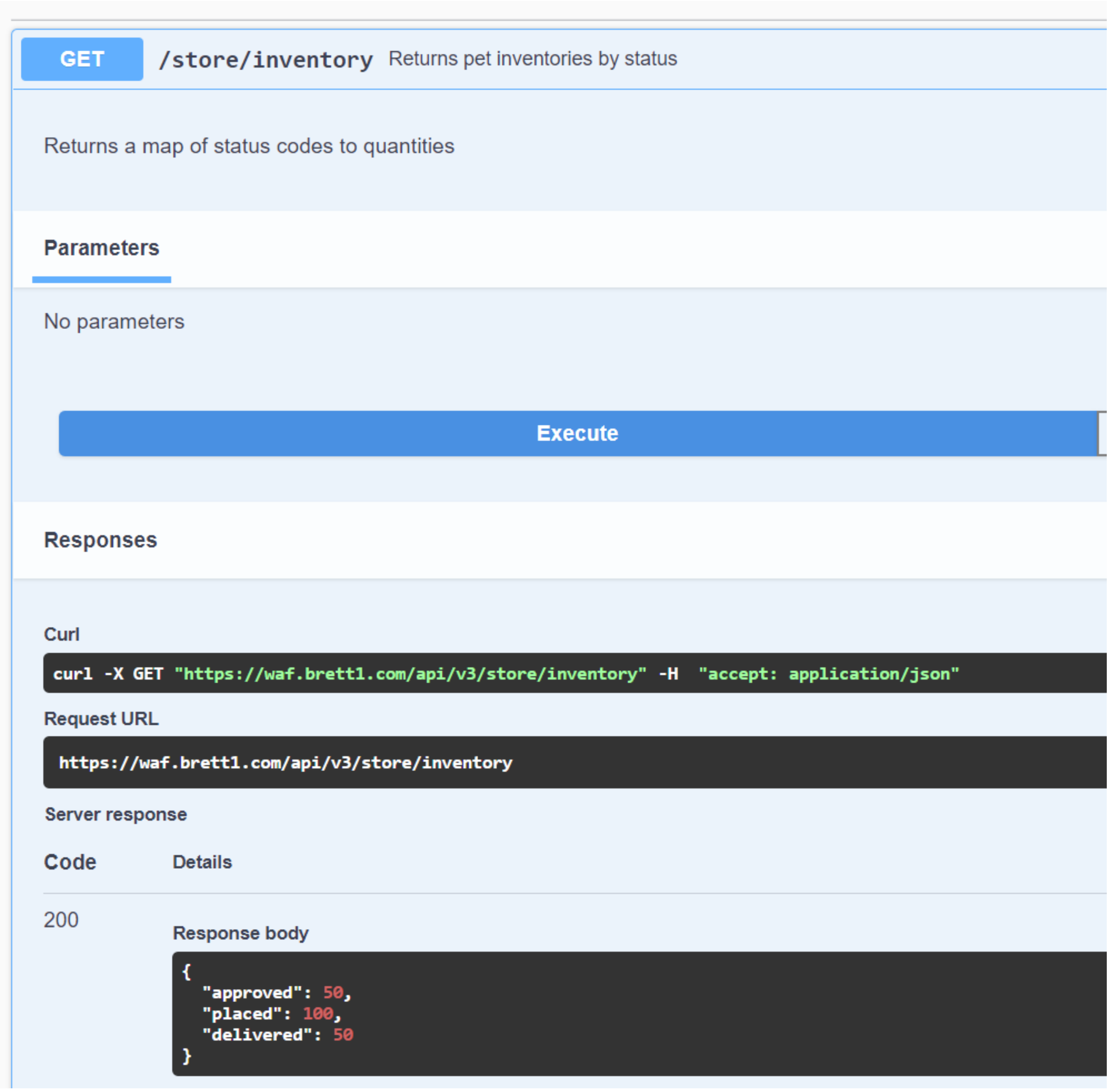
Response headers

```
 access-control-allow-headers: Content-Type,api_key,Authorization
 access-control-allow-methods: GET,POST,DELETE,PUT
 access-control-allow-origin: *
 access-control-expose-headers: Content-Disposition
 content-length: 43
 content-type: application/xml
 date: Fri,16 Apr 2021 14:24:47 GMT
 x-expires-after: Fri Apr 16 15:24:47 GMT 2021
 x-rate-limit: 5000
```

Responses

Code	Description
200	successful operation

9. Scroll to STORE -> /store/inventory click Try it out, then execute, and it looks like this



/store/inventory Returns pet inventories by status

Returns a map of status codes to quantities

Parameters

No parameters

Responses

Curl

```
curl -X GET "https://waf.brett1.com/api/v3/store/inventory" -H  "accept: application/json"
```

Request URL

```
https://waf.brett1.com/api/v3/store/inventory
```

Server response

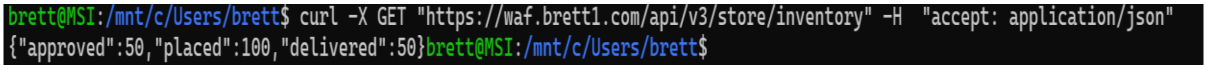
Code Details

200

Response body

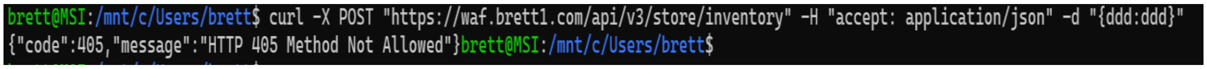
```
{
  "approved": 50,
  "placed": 100,
  "delivered": 50
}
```

10. Copy the CURL command and try it out on your computer, make sure it works.



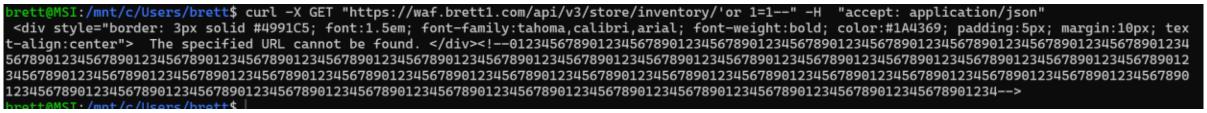
```
brett@MSI:/mnt/c/Users/brett$ curl -X GET "https://waf.brett1.com/api/v3/store/inventory" -H  "accept: application/json"
{"approved":50,"placed":100,"delivered":50}brett@MSI:/mnt/c/Users/brett$
```

11. Then change the method to a POST and try that, and verify the WAF blocks the POST because POST is not allowed in the JSON Schema that you imported for the /store/inventory endpoint

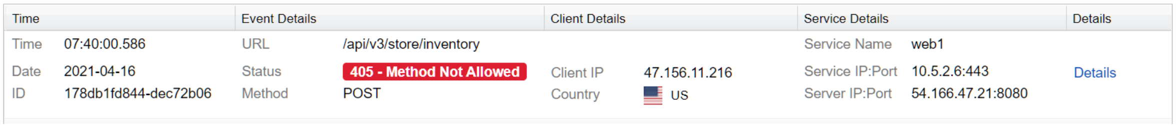


```
brett@MSI:/mnt/c/Users/brett$ curl -X POST "https://waf.brett1.com/api/v3/store/inventory" -H "accept: application/json" -d "{ddd:ddd}"
{"code":405,"message":"HTTP 405 Method Not Allowed"}brett@MSI:/mnt/c/Users/brett$
```

12. Go back to doing a GET but add a SQL injection to the end of the GET verify the WAF blocks this

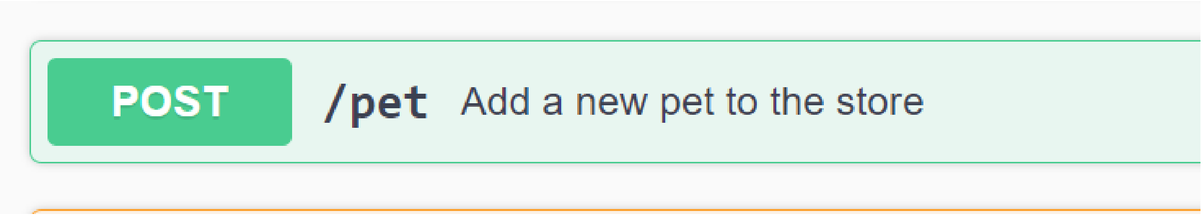


```
brett@MSI:/mnt/c/Users/brett$ curl -X GET "https://waf.brett1.com/api/v3/store/inventory/'or 1=1--" -H  "accept: application/json"
 <div style="border: 3px solid #4991C5; font:1.5em; font-family:tahoma,calibri,arial; font-weight:bold; color:#1A4369; padding:5px; margin:10px; tex
t-align:center">  The specified URL cannot be found. </div><!--01234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234
5678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012
3456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234567890123456789012345678901234-->
brett@MSI:/mnt/c/Users/brett$
```

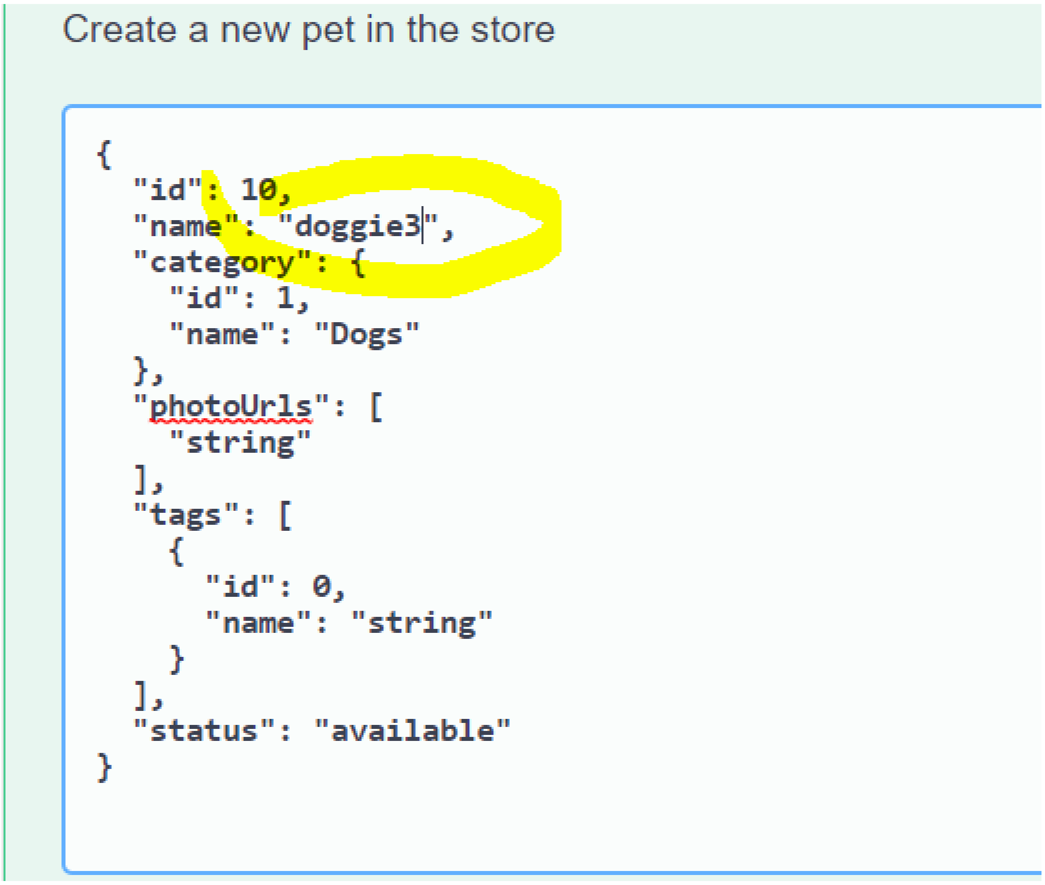


Time		Event Details		Client Details		Service Details		Details
Time	07:40:00.586	URL	/api/v3/store/inventory			Service Name	web1	
Date	2021-04-16	Status	405 - Method Not Allowed	Client IP	47.156.11.216	Service IP:Port	10.5.2.6:443	Details
ID	178db1fd844-dec72b06	Method	POST	Country	US	Server IP:Port	54.166.47.21:8080	

13. Back in the browser, scroll to PET -> add pet to store and click it, then click try it out



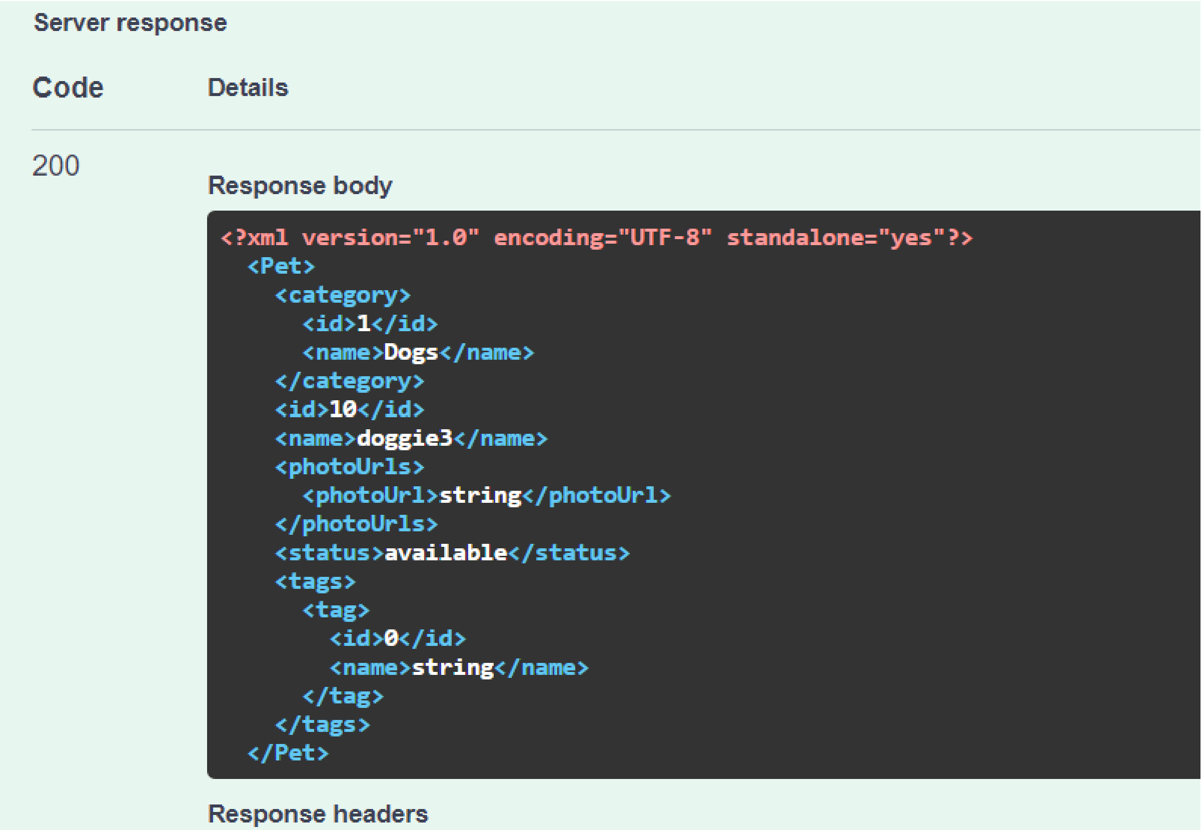
14. Give your pet a unique name and click execute



Create a new pet in the store

```
{
  "id": 10,
  "name": "doggie3",
  "category": {
    "id": 1,
    "name": "Dogs"
  },
  "photoUrls": [
    "string"
  ],
  "tags": [
    {
      "id": 0,
      "name": "string"
    }
  ],
  "status": "available"
}
```

15. Verify you get a json response similar to this



Server response

Code Details

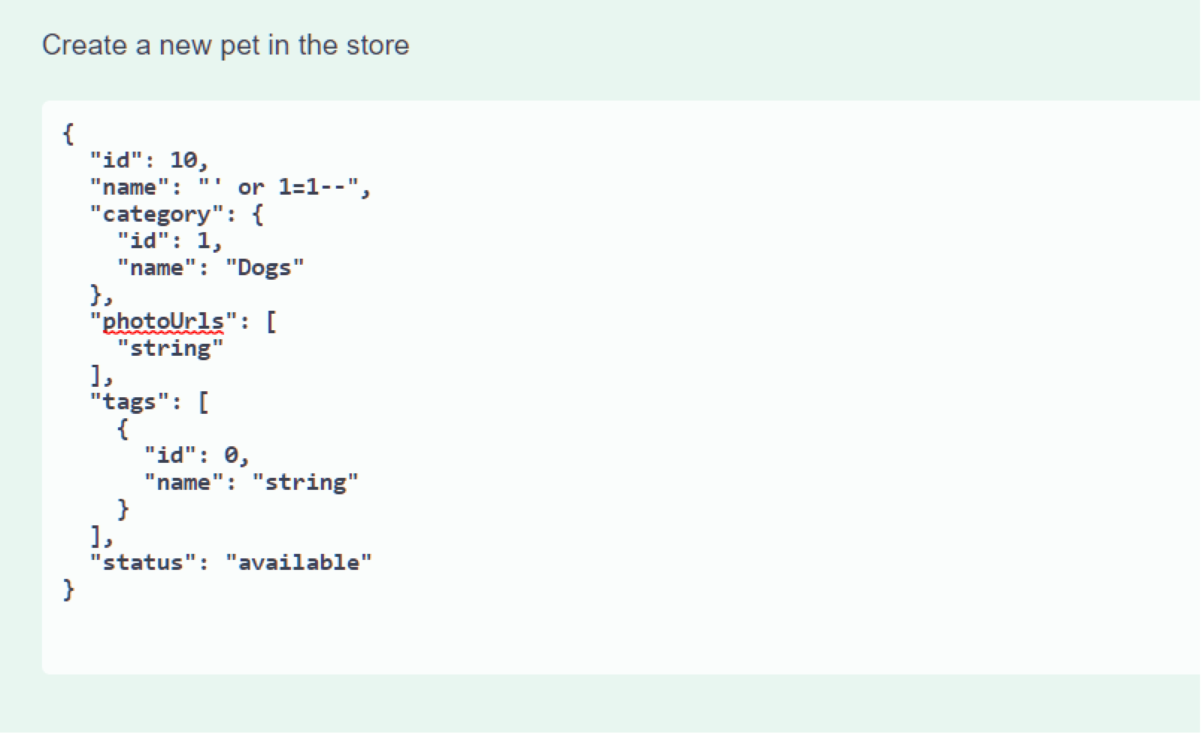
200

Response body

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <Pet>
    <category>
      <id>1</id>
      <name>Dogs</name>
    </category>
    <id>10</id>
    <name>doggie3</name>
    <photoUrls>
      <photoUrl>string</photoUrl>
    </photoUrls>
    <status>available</status>
    <tags>
      <tag>
        <id>0</id>
        <name>string</name>
      </tag>
    </tags>
  </Pet>
```

Response headers

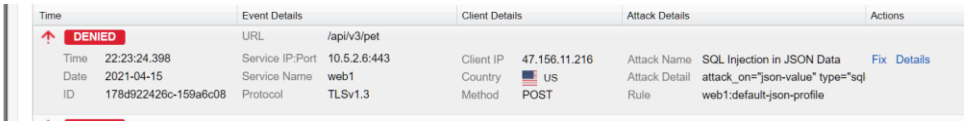
16. Copy the CURL command Change the name to ‘or 1=1– which is s SQL injection and execute



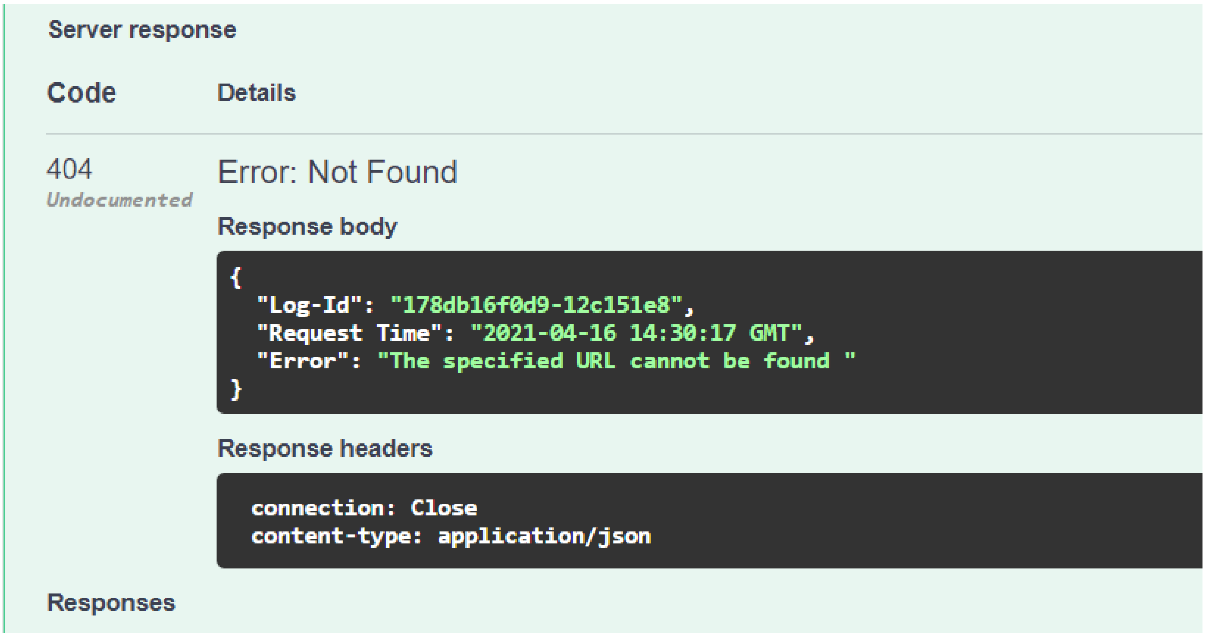
Create a new pet in the store

```
{
  "id": 10,
  "name": "' or 1=1--",
  "category": {
    "id": 1,
    "name": "Dogs"
  },
  "photoUrls": [
    "string"
  ],
  "tags": [
    {
      "id": 0,
      "name": "string"
    }
  ],
  "status": "available"
}
```

17. Verify the WAF blocked this



Time		Event Details		Client Details		Attack Details		Actions
DENIED		URL	/api/v3/pet					
Time	22:23:24.398	Service IP:Port	10.5.2.6:443	Client IP	47.156.11.216	Attack Name	SQL Injection in JSON Data	Fix Details
Date	2021-04-15	Service Name	web1	Country	US	Attack Detail	attack_on="json-value" type="sql	
ID	178d922426c-159a6c08	Protocol	TLSv1.3	Method	POST	Rule	web1:default-json-profile	



Server response

Code	Details
404 *Undocumented*	Error: Not Found

Response body

```
{
  "Log-Id": "178db16f0d9-12c151e8",
  "Request Time": "2021-04-16 14:30:17 GMT",
  "Error": "The specified URL cannot be found "
}
```

Response headers

```
 connection: Close 
 content-type: application/json 
```

Responses

18.