Hands-on Lab - Modernize JPetStore with Microservices

Estimated Time: 30 minutes

In this lab, you will become familiar with using the Swagger UI. The Swagger UI is an open source project to visually render documentation for an API defined with the OpenAPI (Swagger) specification. REST APIs endpoint is one end of a communication channel. When an API interacts with another system, the touchpoints of this communication are considered endpoints. For APIs, an endpoint can include a URL of a server or service.

Learning Objectives:

After completing this exercise, you should be able to perform the following tasks:

- Use the Swagger UI and understand the various components.
- Access endpoints through the Swagger UI.

Pre-requisites

- You must be familiar with Docker applications and commands.
- You must have a good understanding of REST API.

Task 1 - Getting the Swagger UI

1. Open a terminal window by using the top menu in the IDE: **Terminal > New Terminal.**

2. In the terminal, run the following command to pull the docker image for Swagger.

```
1
```

```
docker pull swaggerapi/swagger-ui
```

Copied! Executed!

3. To run the swagger application and access the UI, run the following command.

```
1
```

```
docker run -dp 8080:8080 swaggerapi/swagger-ui
```

Copied! Executed!

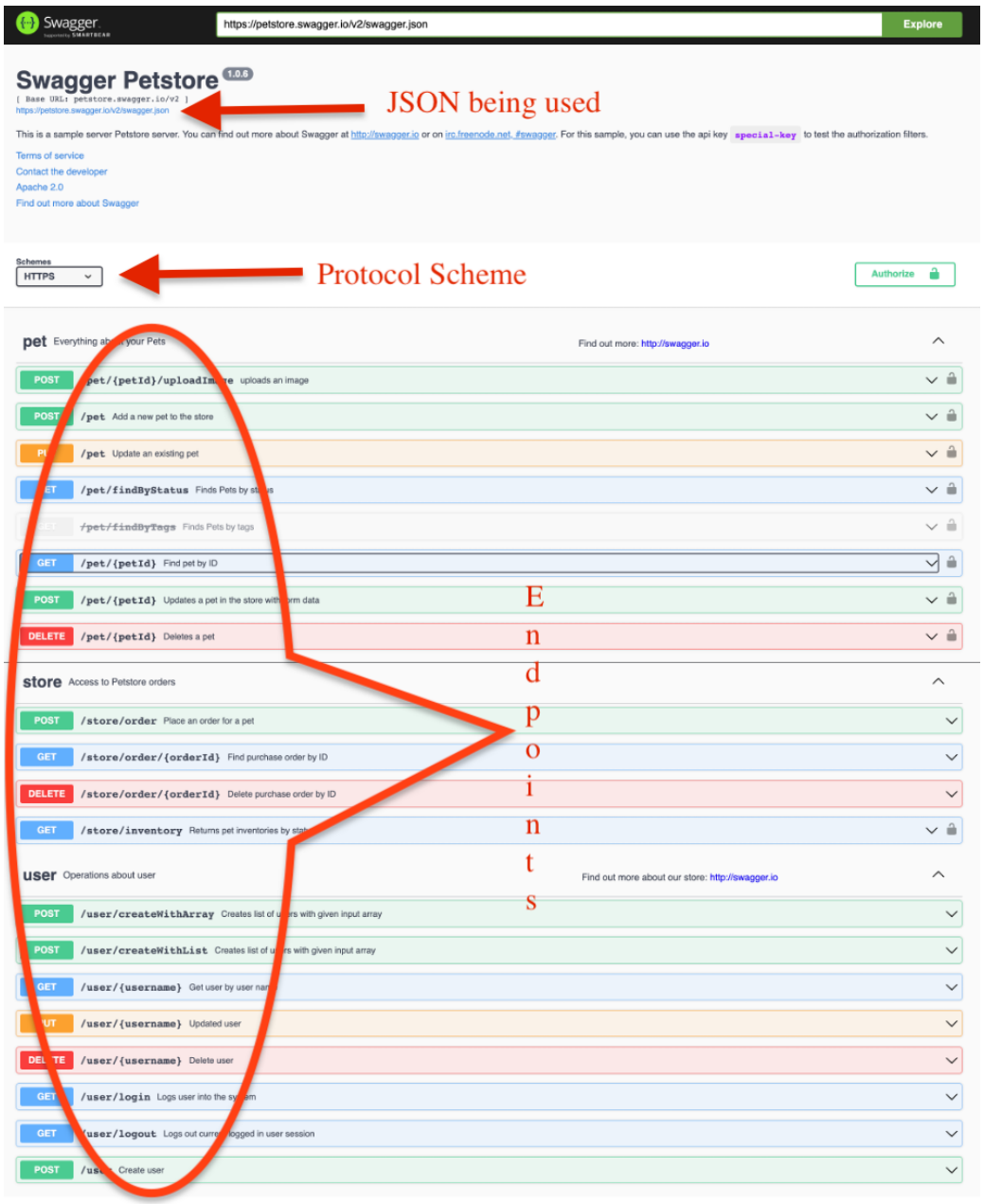
   Note - You will get a hex code in return. This means the server has successfully started.

4. Click on the button below or click the **Skills Network** icon and choose **Launch Application** from the menu and enter the port number *8080*.

Launch Application

5. It opens the browser and connects to port 8080 which is where the Swagger application is running.

Swagger UI comes up connecting to the default, preconfigured sample PetStore JSON.



The page will look as shown below. It lists the following:

1. JSON being used
2. The protocol scheme
3. The end points along with the type of REST requests - GET, POST, PUT, UPDATE, DELETE
4. It also lists the Models that are used in the application

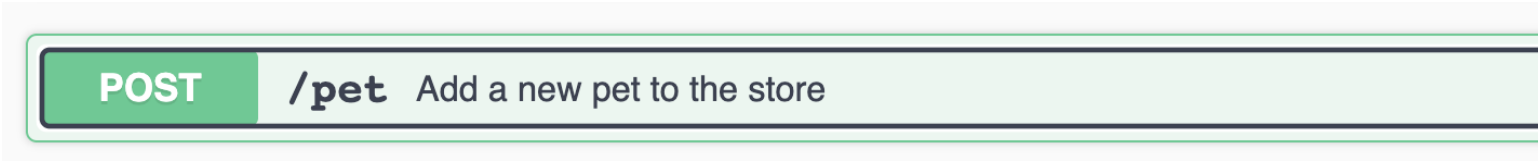
Task 2 - Accessing endpoints through the Swagger UI

As a part of this task you will -

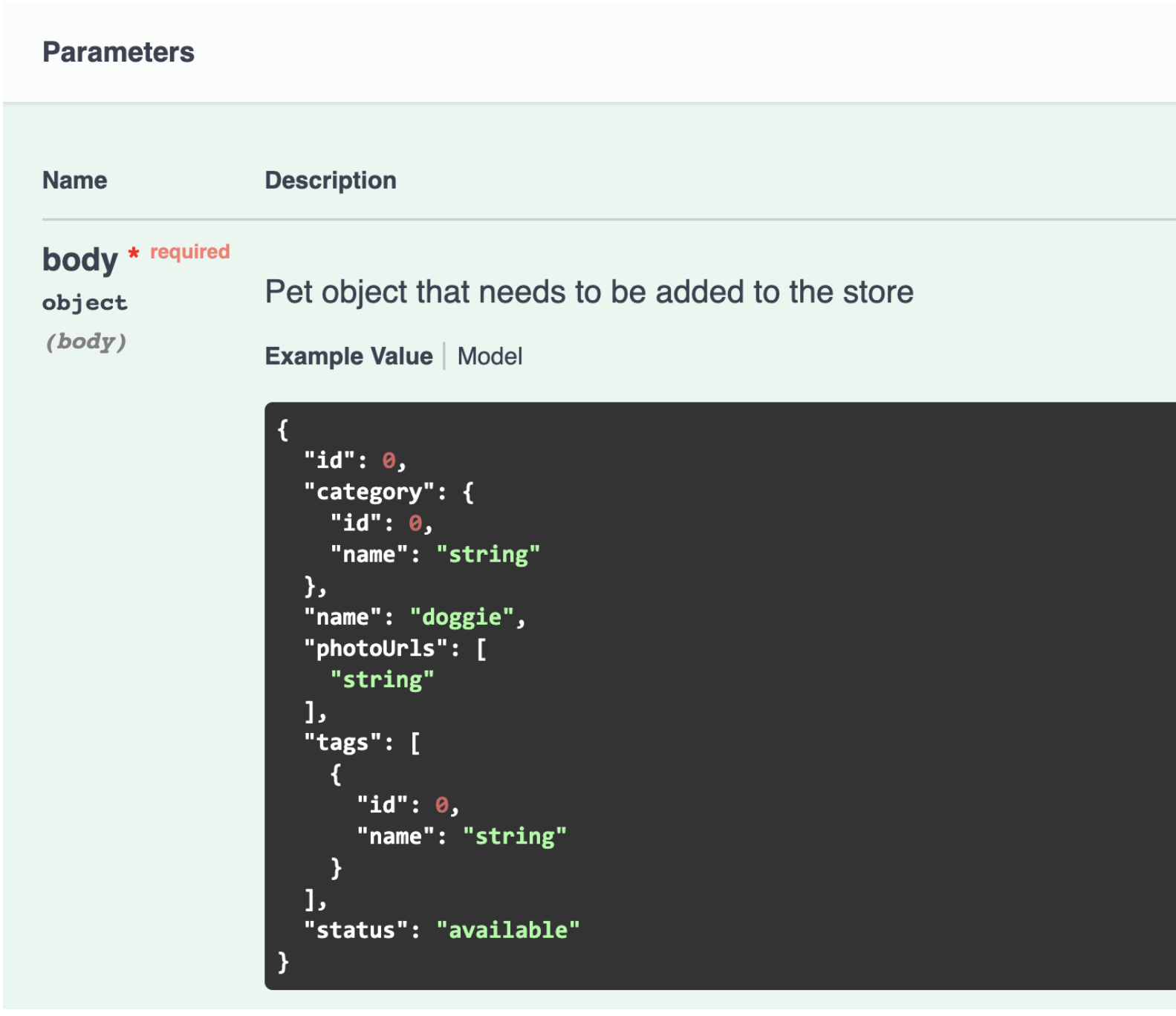
1. Add a pet
2. Get the details of the pet by id
3. Update the pet status to **sold**
4. Get the details of the pet by id to see if the status has been updated
5. Delete the pet
6. Get the details of the pet by id to see that it doesn't exist

You will try POST, GET and DELETE endpoints.

1. Click the dropdown next to the end point **POST /pet**.



2. Click on **Try it out** to add a pet.



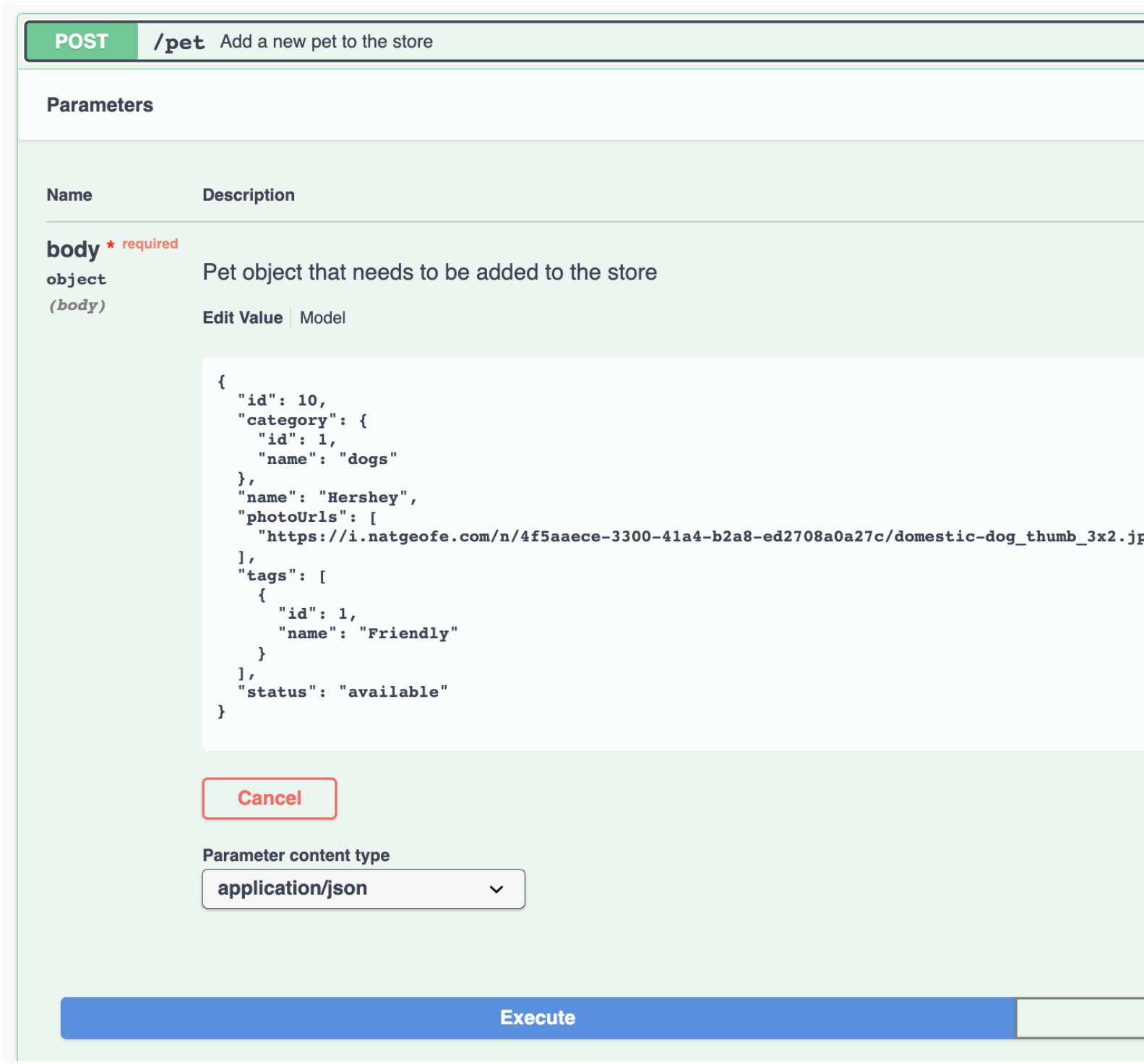
3. This will allow you to edit the values. Replace the prepopulated JSON, with the following:

```
1
2
3
4
5
6
7
8
9
10
11
12
```

```
13. 13
14. 14
15. 15
16. 16
17. 17
18. 18
```

```
1. {
2.   "id": 10,
3.   "category": {
4.     "id": 1,
5.     "name": "dogs"
6.   },
7.   "name": "Hershey",
8.   "photoUrls": [
9.     "https://i.natgeofe.com/n/4f5aaece-3300-41a4-b2a8-ed2708a0a27c/domestic-dog_thumb_3x2.jpg"
10.  ],
11.  "tags": [
12.    {
13.      "id": 1,
14.      "name": "Friendly"
15.    }
16.  ],
17.  "status": "available"
18. }
```

Copied!



4. Click **Execute** to add the pet. You should see a **Server Response** with Code 200, which means the POST request was successful.

Server response

Code	Details
200 *Undocumented*	Response body


```
{
  "id": 10,
  "category": {
    "id": 1,
    "name": "dogs"
  },
  "name": "Hershey",
  "photoUrls": [
    "https://i.natgeofe.com/n/4f5aaece-3300-41a4-b2a8-ed2708a0a27c/domestic-dog_thumb_3x2.jpg"
  ],
  "tags": [
    {
      "id": 1,
      "name": "Friendly"
    }
  ],
  "status": "available"
}
```

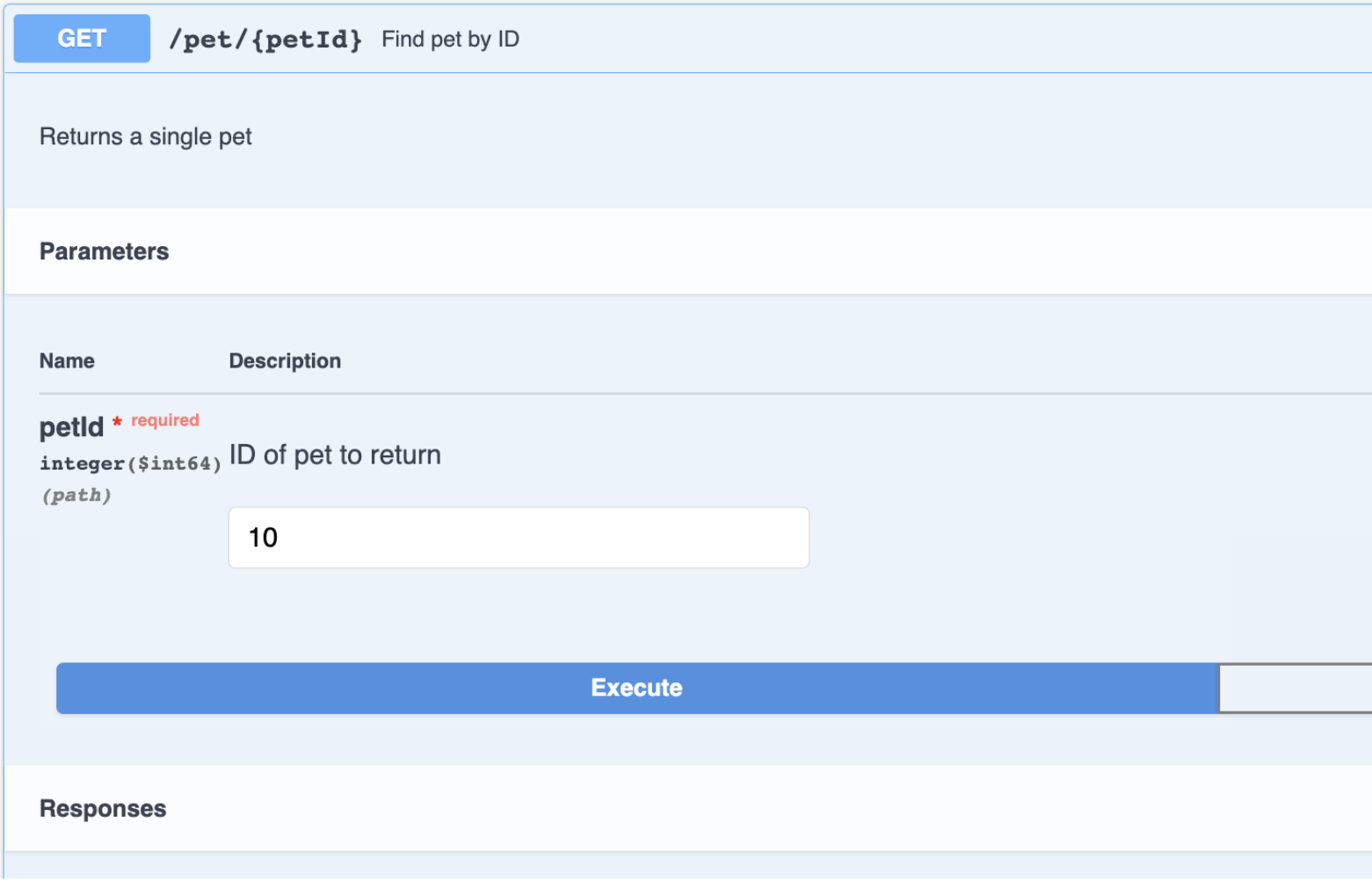

Response headers

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

5. Close the dropdown for **POST /pet**.

6. Now you will get the details of the pet you just added. Click the dropdown next to **GET /pet/{petid}**.

7. Click **Try it out** and enter the id of the pet whose details you want to retrieve. Our pet's id is **10**. Click **Execute**.



8. In the Server Response obtained, you will see **200** indicating the request was successful and the details of the pet.

Responses

Curl

```
curl -X 'GET' \
  'https://petstore.swagger.io/v2/pet/10' \
  -H 'accept: application/json' \
  -H 'api_key: special-key'
```

Request URL

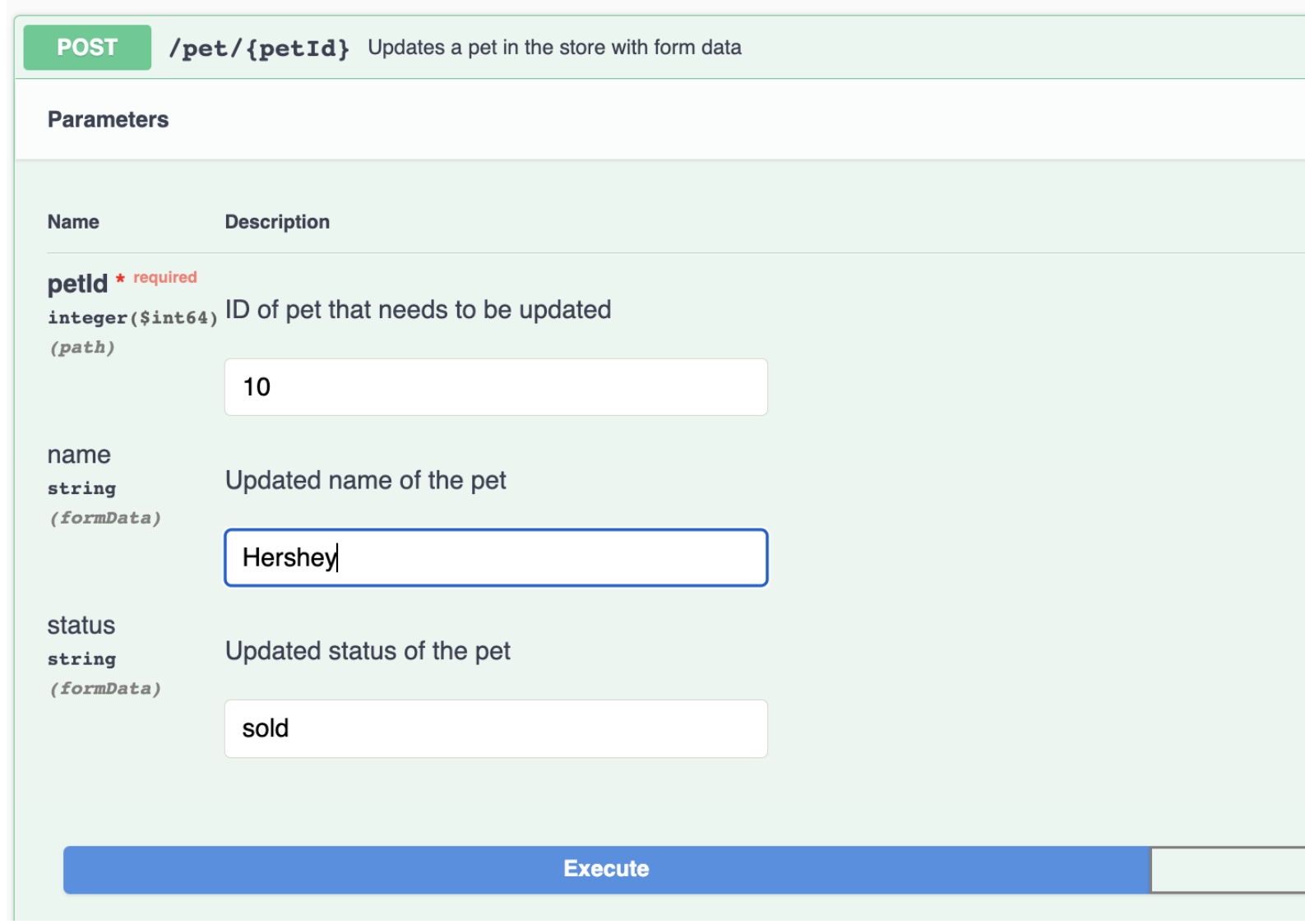
```
https://petstore.swagger.io/v2/pet/10
```

Server response

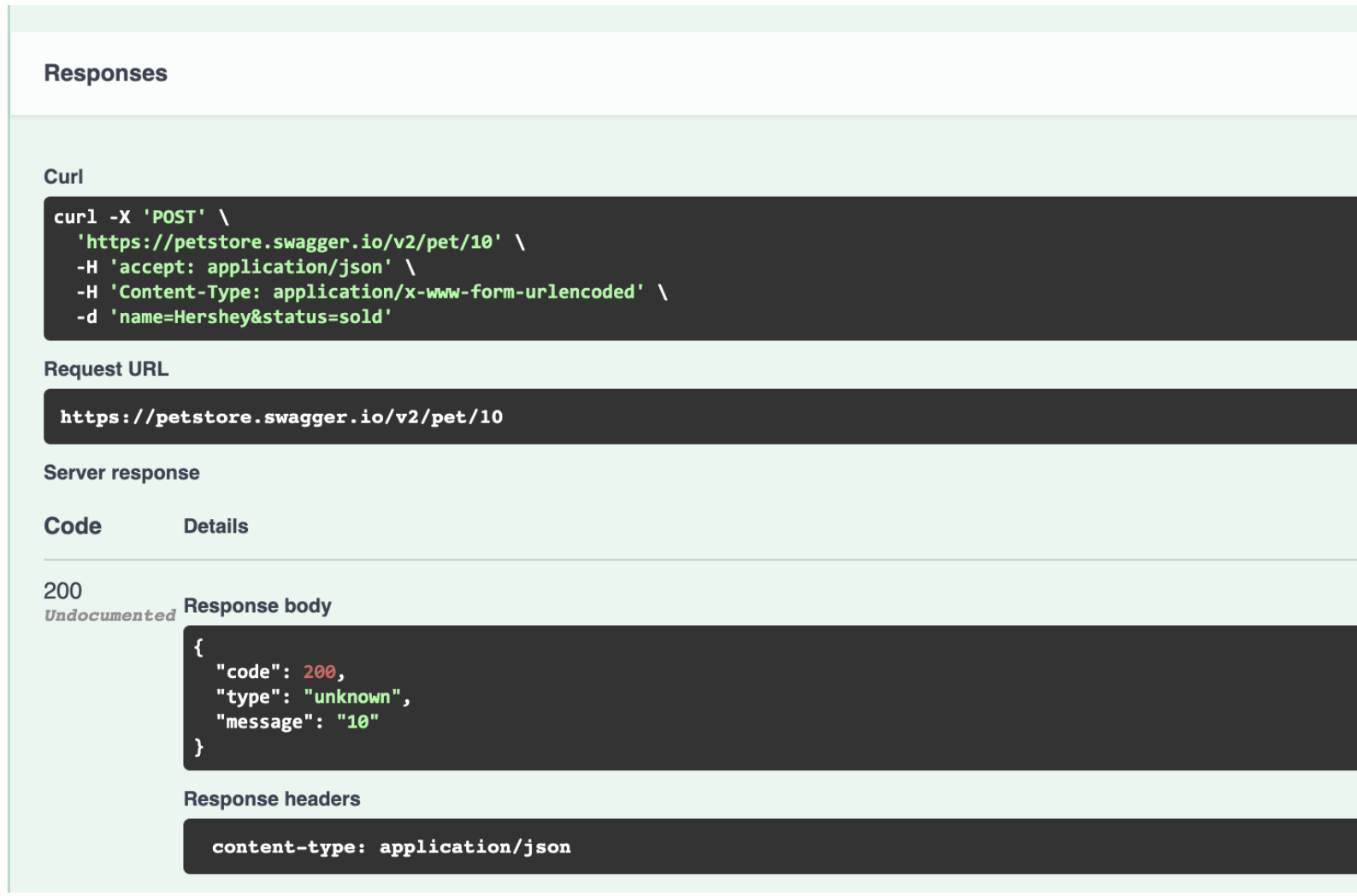
Code	Details
200	**Response body**


```
{
  "id": 10,
  "category": {
    "id": 1,
    "name": "dogs"
  },
  "name": "Hershey",
  "photoUrls": [
    "https://i.natgeofe.com/n/4f5aaece-3300-41a4-b2a8-ed2708a0a27c/domestic-dog_thumb_3x2.jpg"
  ],
  "tags": [
    {
      "id": 1,
      "name": "Friendly"
    }
  ],
  "status": "available"
}
```


9. Close the **GET /pet/{petid}** dropdown.
10. You will now update the status of the pet to **sold**. To do this, click the dropdown next to **POST /pet/{petid}**.
11. Now you will change the status of the pet with id **10** to **sold**. Click **Try it out** and make the changes as shown below, and then click **Execute**.



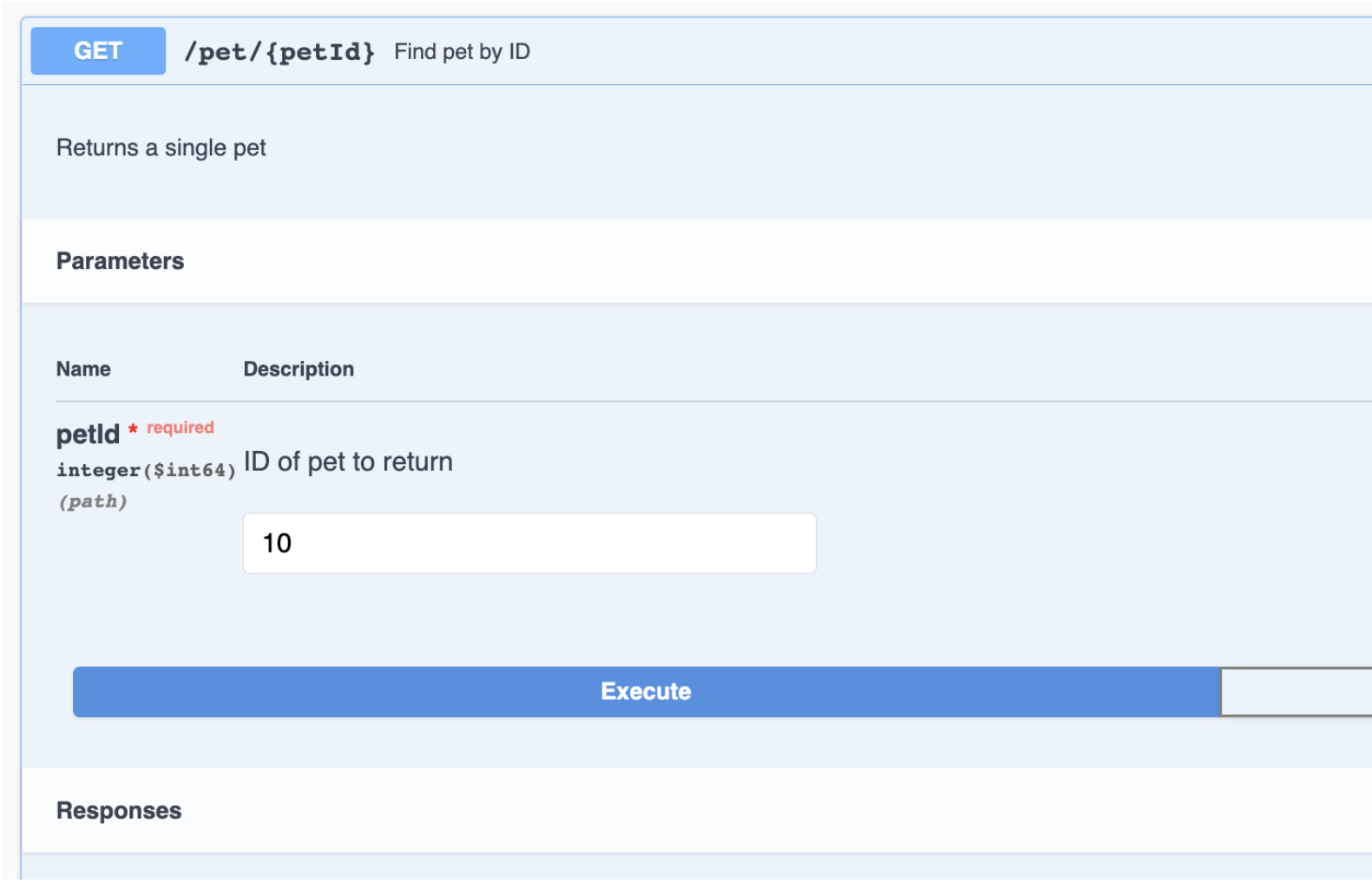
12. If the update ran successfully, you get a server response with code **200**.



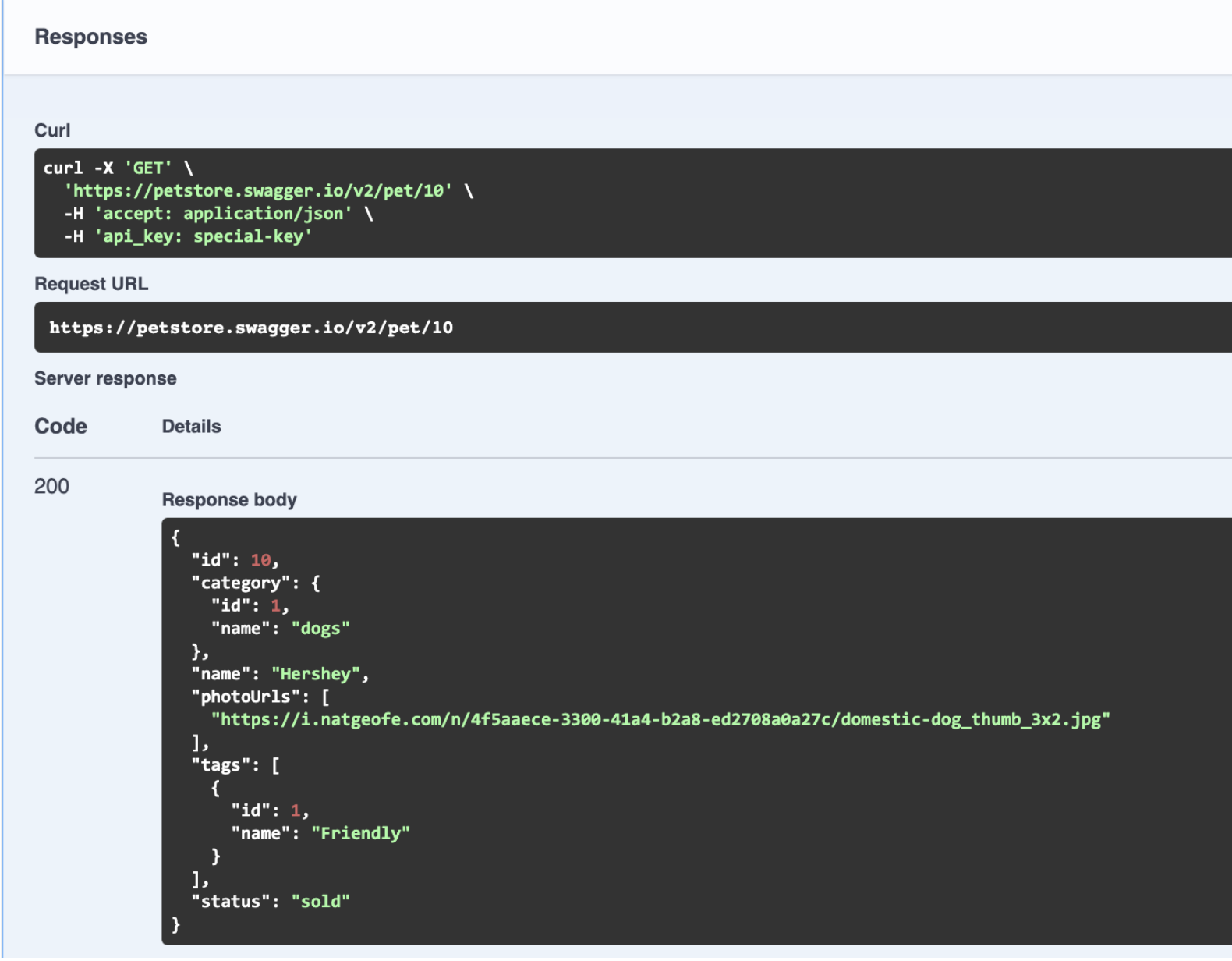
13. Close the **POST /pet/{petid}** dropdown.

14. You can check if the pet you just updated reflects as **sold**. Click the dropdown next to **GET /pet/{petid}**.

15. Enter the id of the pet whose details you want to retrieve. The pet you updated has the id **10**. Click **Execute**.

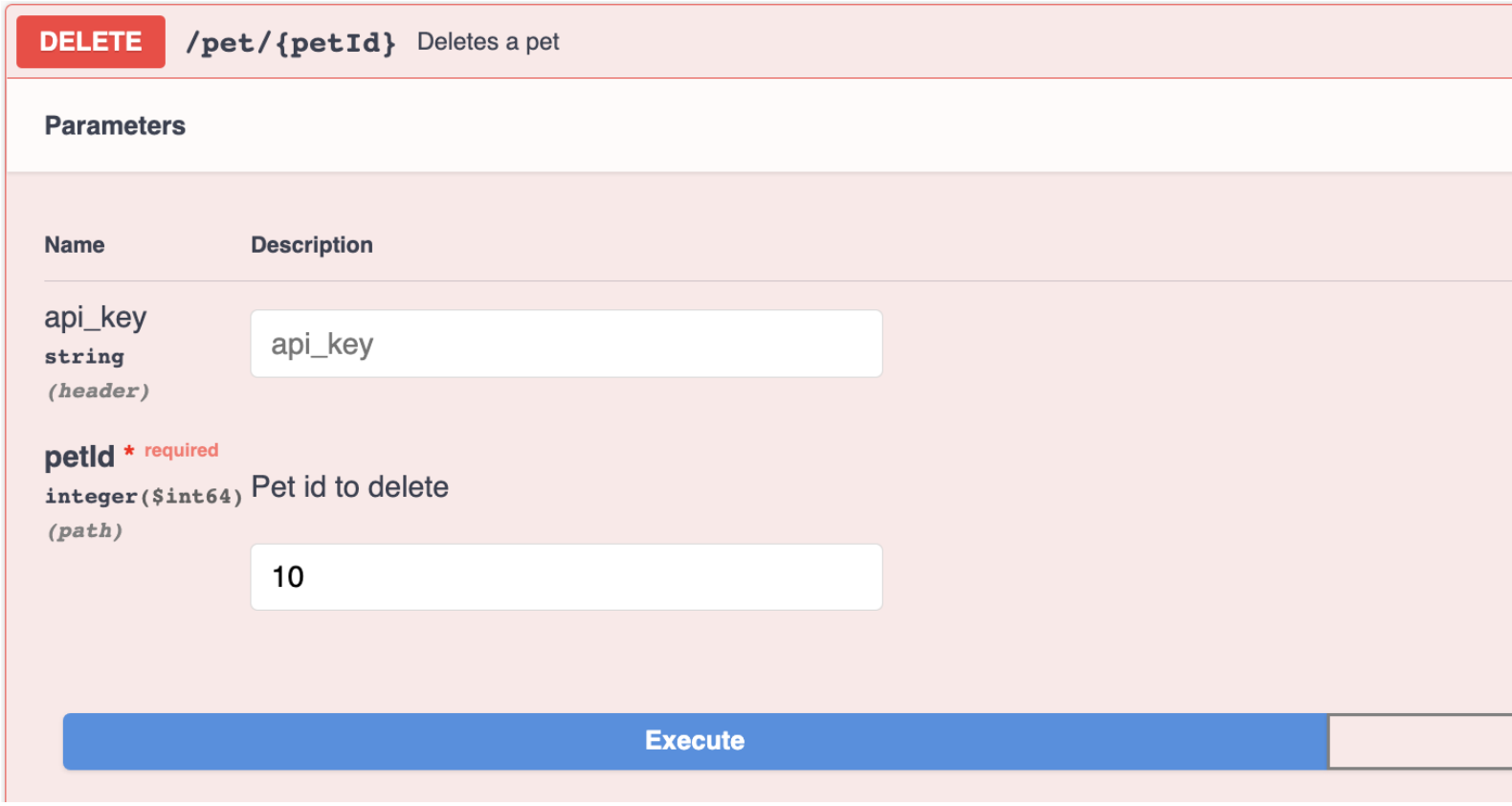


16. In the Server Response obtained, you will see **200** indicating the request was successful and the details of the pet.



17. Close the **GET /pet/{petid}** dropdown.

18. Now that the pet is sold, you can delete it from your system. Click the dropdown next to **DELETE /pet/{petid}**. Click **Try it out** and enter the **petid** as 10.



19. Click **Execute**. If the delete ran successfully, you get a server response with code **200**.

Responses

Curl

```
curl -X 'DELETE' \
  'https://petstore.swagger.io/v2/pet/10' \
  -H 'accept: application/json'
```

Request URL

```
https://petstore.swagger.io/v2/pet/10
```

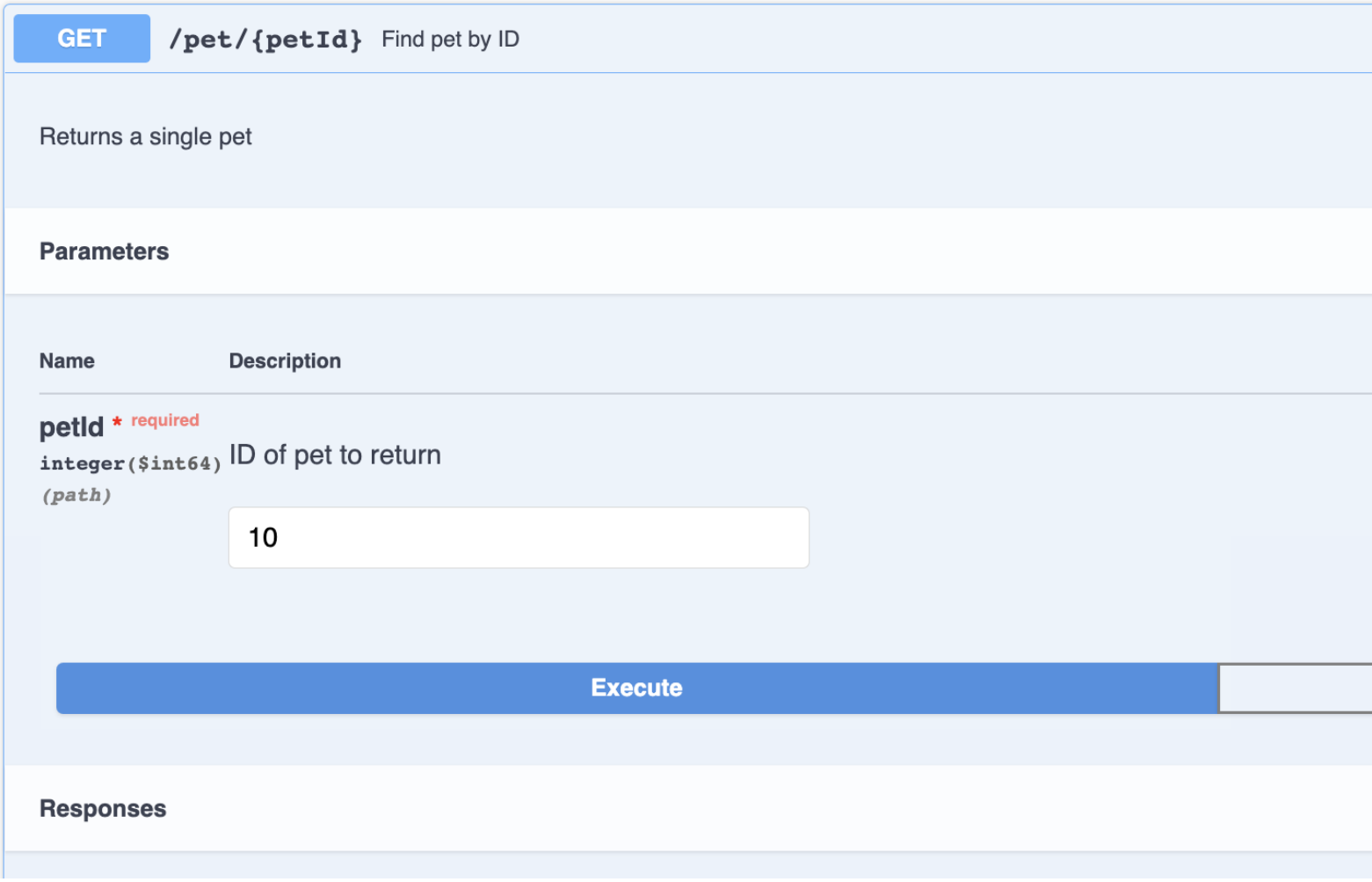
Server response

Code	Details
200 *Undocumented*	**Response body**

```
{
  "code": 200,
  "type": "unknown",
  "message": "10"
}
```

20. You can check if the pet you just deleted, has been removed from the system. Click the dropdown next to **GET /pet/{petid}**.

21. Enter the id of the pet whose details you want to retrieve. The pet you deleted has the id **10**. Click **Execute**.



22. In the Server Response obtained you will see **404** indicating the request was errorneous and there is no such pet.

Congratulations! You have successfully completed the task.

Tutorial details

Author: Lavanaya T S

Contributors: Pallavi Rai

Change Log

Date	Version	Changed by	Change Description
2022-08-22	1.0	Lavanaya T S	Initial version created
2022-11-25	1.1	Steve Hord	QA pass edits
2023-09-05	1.2	Sapthashree	Updated the docker run command