

Temuan SQLi

1. Sql Injection

The screenshot shows a 'Login' form with two input fields. The 'Email*' field contains the value "' OR 1=1 --". The 'Password*' field contains a series of dots ('.....'). Below the form is a link 'Lupa password Anda?' and a large blue 'Login' button. There is also a checkbox labeled 'Ingatkan saya'.

The screenshot shows the OWASP Juice Shop application. At the top, there is a navigation bar with links for 'Account', 'Orders & Payment', 'Privacy & Security', and 'Logout'. A success message in a green box says: 'Server sudah direstart: Your previous hacking progress has been restored automatically.' and 'Anda berhasil memecahkan tantangan: Login Admin (Log in with the administrator's user account.)'. Below this, there is a section titled 'Semua Produk' (All Products) displaying items like 'Apple Juice (1000ml)' and 'Banana Juice (1000ml)'. A user menu is open, showing options such as 'Privacy Policy', 'Request Data Export', 'Request Data Erasure', 'Ganti kata sandi', '2FA Configuration', and 'Last Login IP'.

Penyebab

Pada halaman login, bisa dilakukan SQL Injection di form username dan passwordnya, hal ini dikarenakan string concatenation.

Rekomendasi

Untuk pencegahannya bisa menggunakan Parameterize query

2. Password ga dienkrypt

```
{ "email": "skay.v13@gmail.com", "password": "12345678"}  
email: "skay.v13@gmail.com"  
password: "12345678"
```

Ketika login Email dan Passwordnya tidak di enkrypt

Penyebab

Password tidak di enkrypt

Rekomendasi

Password di enkrypt

3. Broken Access Control (BAC)

```
{ "status": "success",  
  "data": {  
    "id": 2,  
    "coupon": null,  
    "UserId": 2,  
    "createdAt": "2025-11-06T02:04:19.495Z",  
    "updatedAt": "2025-11-06T02:04:19.495Z",  
    "Products": [  
      {  
        "id": 4,  
        "name": "Raspberry Juice (1000ml)",  
        "description": "Made from blended Raspberry Pi, water and sugar.",  
        "price": 4.99,  
        "deluxePrice": 4.99,  
        "image": "raspberry_juice.jpg",  
        "createdAt": "2025-11-06T02:04:19.158Z"...
```

The screenshot shows a REST client interface with the following details:

- URL:** `http://localhost:32768/rest/basket/1`
- Method:** GET
- Headers:** (20)
- Body:** (JSON) - The response body is displayed as follows:

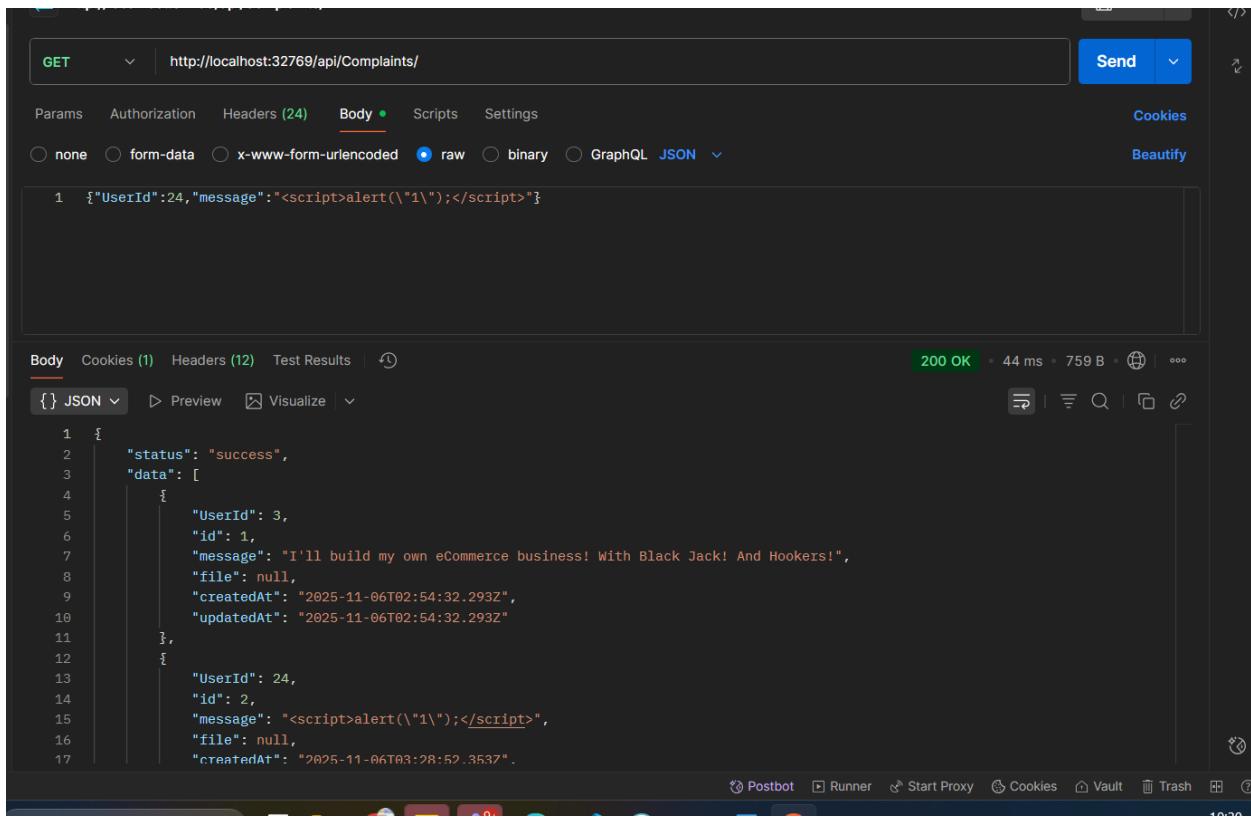
```
1 {  
2     "status": "success",  
3     "data": {  
4         "id": 1,  
5         "coupon": null,  
6         "UserId": 1,  
7         "createdAt": "2025-11-06T02:04:19.495Z",  
8         "updatedAt": "2025-11-06T02:04:19.495Z",  
9         "Products": [  
10             {  
11                 "id": 1,  
12                 "name": "Apple Juice (1000ml)",  
13                 "description": "The all-time classic.",  
14                 "price": 1.99,  
15                 "deluxePrice": 0.99,  
16                 "image": "apple_juice.jpg",  
17                 "createdAt": "2025-11-06T02:04:19.157Z"  
18             }  
19         ]  
20     }  
21 }
```
- Response Status:** 200 OK
- Time:** 80 ms
- Size:** 814 B

Penyebab

Dengan memakai token yang sama, bisa memanggil keranjang milik orang lain, karena tidak ada validasi akses kontrolnya

Rekomendasi

Sistem harus memvalidasi terhadap keranjang dari requesternya



GET http://localhost:32769/api/Complaints/

Params Authorization Headers (24) Body Scripts Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

1 {"UserId":24,"message":<script>alert(\"1\");</script>}

Body Cookies (1) Headers (12) Test Results

200 OK 44 ms 759 B

{ } JSON Preview Visualize

```
1 {
  "status": "success",
  "data": [
    {
      "UserId": 3,
      "id": 1,
      "message": "I'll build my own eCommerce business! With Black Jack! And Hookers!",
      "file": null,
      "createdAt": "2025-11-06T02:54:32.293Z",
      "updatedAt": "2025-11-06T02:54:32.293Z"
    },
    {
      "UserId": 24,
      "id": 2,
      "message": "<script>alert(\"1\");</script>",
      "file": null,
      "createdAt": "2025-11-06T03:28:52.353Z"
    }
  ]
}
```

Pada menu complaints Ketika mensubmit complain yg method POST, saya ubah method nya menjadi GET. Saya bisa melihat complain dari semua user

Penyebab

Tidak ada validasi pada Method GET Complaint

Rekomendasi

Methodnya harus divalidasi bahwa hanya admin saja yg boleh lihat