

1. Login Bypass sebagai admin (SQL Injection)

The screenshot shows the OWASP Juice Shop login interface. The 'Email*' field contains the value 'or 1=1--'. The 'Password*' field contains a single dot ('.') and has an eye icon for password visibility.

The screenshot shows the OWASP Juice Shop dashboard. The user is logged in as 'admin@juice-shop'. The navigation bar includes links for 'Orders & Payment' (with a red notification badge), 'Privacy & Security', and 'Logout'. The main content area displays 'All Products' with three items: 'Apple Juice (1000ml)' at 1.99, 'Apple Pomace' at 0.89, and another item partially visible.

Penyebab

Aplikasi menyusun kueri autentikasi dengan menggabungkan string dari input:

```
{  
  "email": "'or 1=1--'",  
  "password": "1"  
}
```

Operator OR 1=1 selalu bernilai benar, sedangkan -- mengomentari sisa kueri sehingga pemeriksaan sandi terlewati. Sistem kemudian menganggap autentikasi berhasil (sering kali terhadap baris pertama/akun berhak tinggi).

Tidak ada parameterisasi: input langsung ditempel ke kueri.

Validasi input lemah: format email/panjang tidak dibatasi.

Logika autentikasi rapuh: “ada baris hasil” dianggap setara “berhasil login”.

Rekomendasi Perbaikan

Tingkat Desain

- Terapkan prinsip “data ≠ perintah” melalui parameterized queries / prepared statements pada semua akses DB.
- Rancang alur autentikasi: temukan akun berdasarkan email → verifikasi hash sandi (bcrypt/Argon2) → constant-time compare → buat sesi/token.
- Terapkan least privilege untuk kredensial DB aplikasi.

Tingkat Implementasi

- Gunakan placeholder terikat parameter:
- Hash sandi (bcrypt/Argon2); jangan pernah menyimpan sandi dalam teks jelas.
- Rate limiting + lockout atas percobaan gagal; pesan kesalahan generik (hindari bocor detail).
- Logging & monitoring untuk pola input mencurigakan.

Tingkat Uji

- Uji negatif dengan payload klasik (' OR 1=1--) dan variasi lainnya.
- Integrasikan SAST/DAST dan security unit tests untuk rute autentikasi.

2. Mass Update Semua Ulasan (NoSQL Injection)

Endpoint review menerima objek JSON dari klien dan menggunakan langsung sebagai filter/operasi basis data. Pelaku menyisipkan operator kueri khusus NoSQL:

```
{ "id": { "$ne": -1 }, "message": "NoSQL Injection!" }
```

Operator \$ne berarti “tidak sama dengan”. Karena hampir semua dokumen memiliki id ≠ -1, filter ini mencocokkan seluruh koleksi, sehingga terjadi pembaruan massal.

Penyebab

- Kepercayaan berlebihan pada bentuk JSON klien: server menerima operator seperti \$ne, \$gt, dll.
- Tidak ada pembatasan kepemilikan/otorisasi: filter tidak dipersempit ke sumber daya milik pengguna.
- Penggunaan operasi massal (updateMany) untuk kasus yang seharusnya per-dokumen.
- Validasi/skema longgar: tidak ada pembatasan tipe, panjang, maupun struktur field.

Rekomendasi Perbaikan

- Terapkan **otorisasi berbasis kepemilikan** (mis. { id: reviewId, userId: req.user.id }).
- Whitelist field yang boleh diisi; blokir key berbahaya (prefiks \$, karakter .).
- Validasi skema (JSON Schema/Mongoose): tipe, panjang maksimal, enum, pola.
- Sanitasi input dan nonaktifkan interpretasi operator dari klien (mis. konfigurasi strict mode).
- Kredensial DB dengan least privilege (tanpa izin update many bila tidak diperlukan).

TASK B

1. Admin Section (vertical privilege escalation via forced browsing)

The screenshot shows the OWASP Juice Shop administration interface. At the top, there's a green banner stating "You successfully solved a challenge: Admin Section (Access the administration section of the store.)". Below this, the main area is divided into two sections: "Registered Users" and "Customer Feedback".

Registered Users:

- admin@juice-sh.op
- jim@juice-sh.op
- bender@juice-sh.op
- bjoern.kimmich@gmail.com
- ciso@juice-sh.op
- support@juice-sh.op
- morty@juice-sh.op
- mc.safesearch@juice-sh.op
- J12934@juice-sh.op
- wurstbrot@juice-sh.op

Customer Feedback:

- 1 I love this shop! Best products in town! Highly recommended! (**@juice-sh.op) ★★★
- 2 Great shop! Awesome service! (**@juice-sh.op) ★★★
- 3 Nothing useful available here! (**der@juice-sh.op) ★
- 21 Please send me the juicy chatbot NFT in my wallet at /juicy-nft : "purpose betray marriage blame crunch monitor spin slide donate sport lift clutch" (**ereum@juice-sh.op) ★
- Incompetent customer support! Can't even upload photo of broken purchase! Support Team: Sorry, only order confirmation PDFs can be attached to complaints! (anonymous) ★★
- This is the store for awesome stuff of all kinds! (anonymous) ★★
- Never gonna buy anywhere else from now on! Thanks for the great service! (anonymous) ★★

Eksplorasi:

“Paksa” buka rute admin yang disembunyikan (tidak ada tautan di UI) — misalnya menebak/membaca rute di berkas JS lalu akses langsung `/#/administration`. Ini tipe forced browsing.

Root cause:

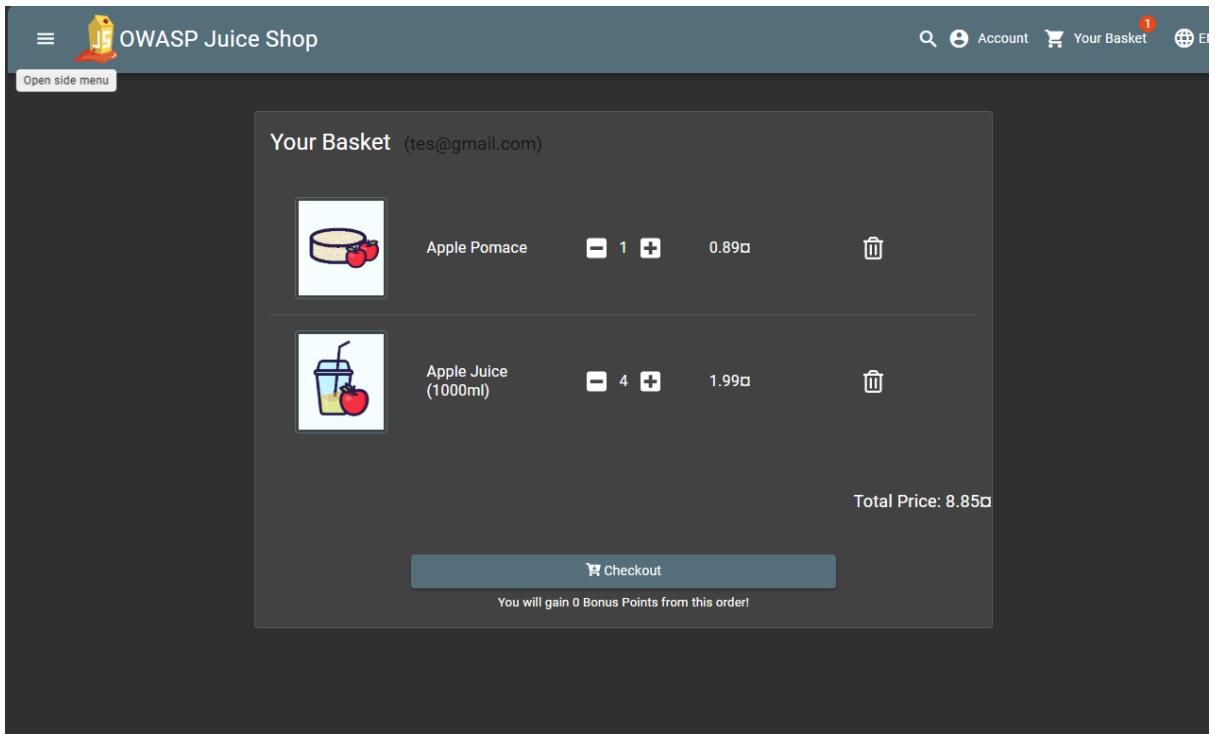
Mengandalkan “security by obscurity”: halaman tidak ditautkan, tetapi tidak ada pemeriksaan otorisasi server-side yang memadai untuk mencegah akses langsung ke rute/fitur admin.

Perbaikan:

Terapkan otorisasi server-side di setiap rute & API (RBAC/ABAC) — sembunyikan tautan ≠ kontrol akses.

Return 403 bila bukan admin, dan pastikan API backend untuk aksi admin juga terlindung

2. Manipulate Basket



Eksplorasi:

Saat menambah barang, intersep request penambahan item (POST/PUT ke endpoint basket items), lalu ganti/selipkan basketId milik korban. Pada beberapa versi/workflow, memodifikasi urutan parameter/field bisa membuat validasi keliru sehingga item ikut masuk ke keranjang korban.

Root cause:

Tidak ada pemeriksaan hak milik di server terhadap basketId yang dikirim klien; validasi logika permintaan tidak konsisten sehingga update “menyasar” resource milik pengguna lain.

Perbaikan:

- Bangun filter di server (gunakan basketId dari sesi pengguna, bukan dari body klien).
- Terapkan object-level checks sebelum mutasi (mis. assert(basket.owner == current_user)).
- Matikan jalur mass-update dari klien; gunakan updateOne dengan guard yang ketat.