# The €666 Checkout That Cost €0

Breaking coupon uniqueness through ordinary cart actions

**Umair Aziz** | October 2026 | Web / business logic

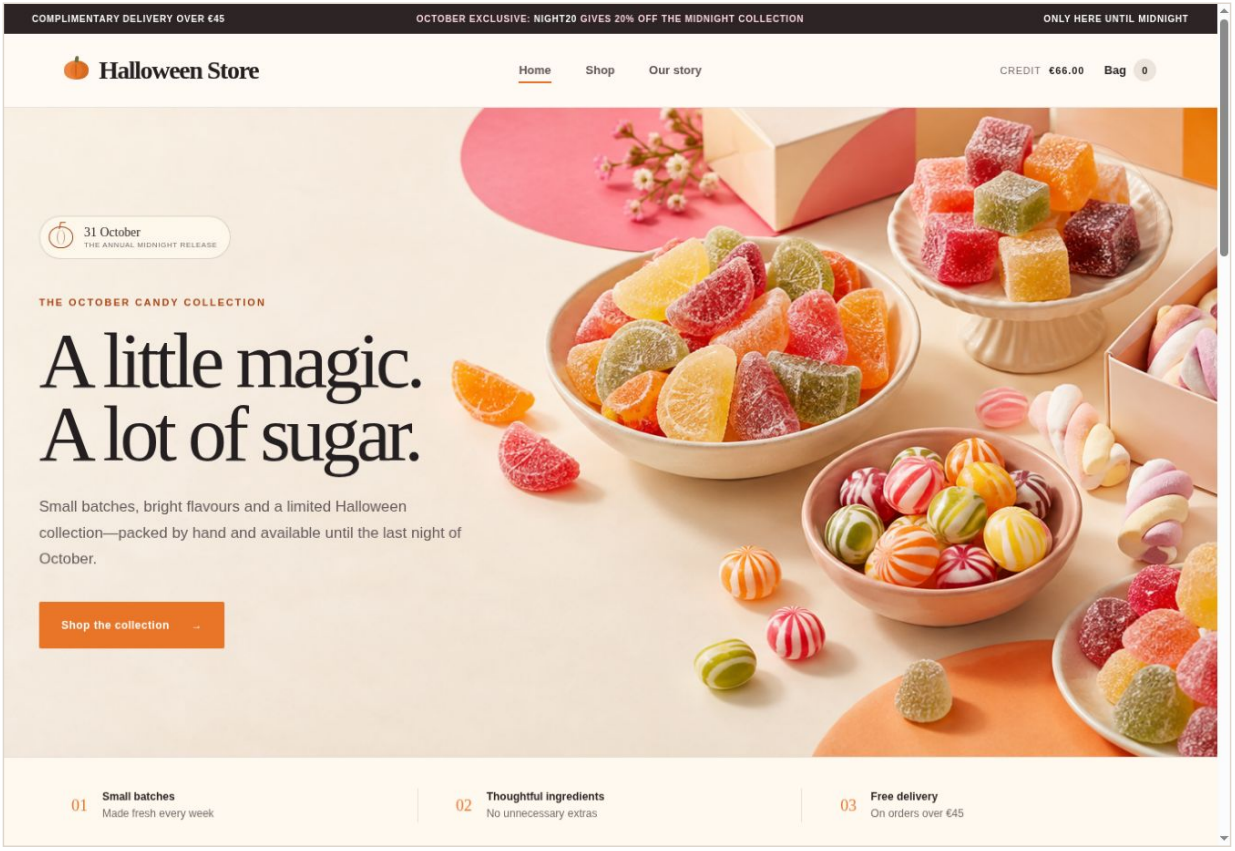



Figure 1. Original storefront: the NIGHT20 banner and €66.00 credit are visible. Route: /challenge.

I started with €66 in store credit and a €666 target item. Applying the two advertised or discoverable coupons once left €333 to pay. By changing the coupon sequence, I made NEWS30 accumulate three times and completed the challenge checkout for €0.

> **Verified outcome**
> NEWS30 × 3 + NIGHT20 × 1 → €0.00 paid.
> Fresh receipt: **BLE-D3642A2C33**. The flag matched my original notes.

Evidence note: these are genuine screenshots from a fresh browser replay on 22 October 2026. Captions identify cropped details; the original images accompany the write-up. No page text or receipt values were reconstructed.

# I found the target and a second coupon

The Midnight Collection was priced at €666.00. The page header showed only €66.00 in credit, while the banner disclosed NIGHT20. I then used the newsletter form with the test address researcher@example.com; it returned NEWS30.

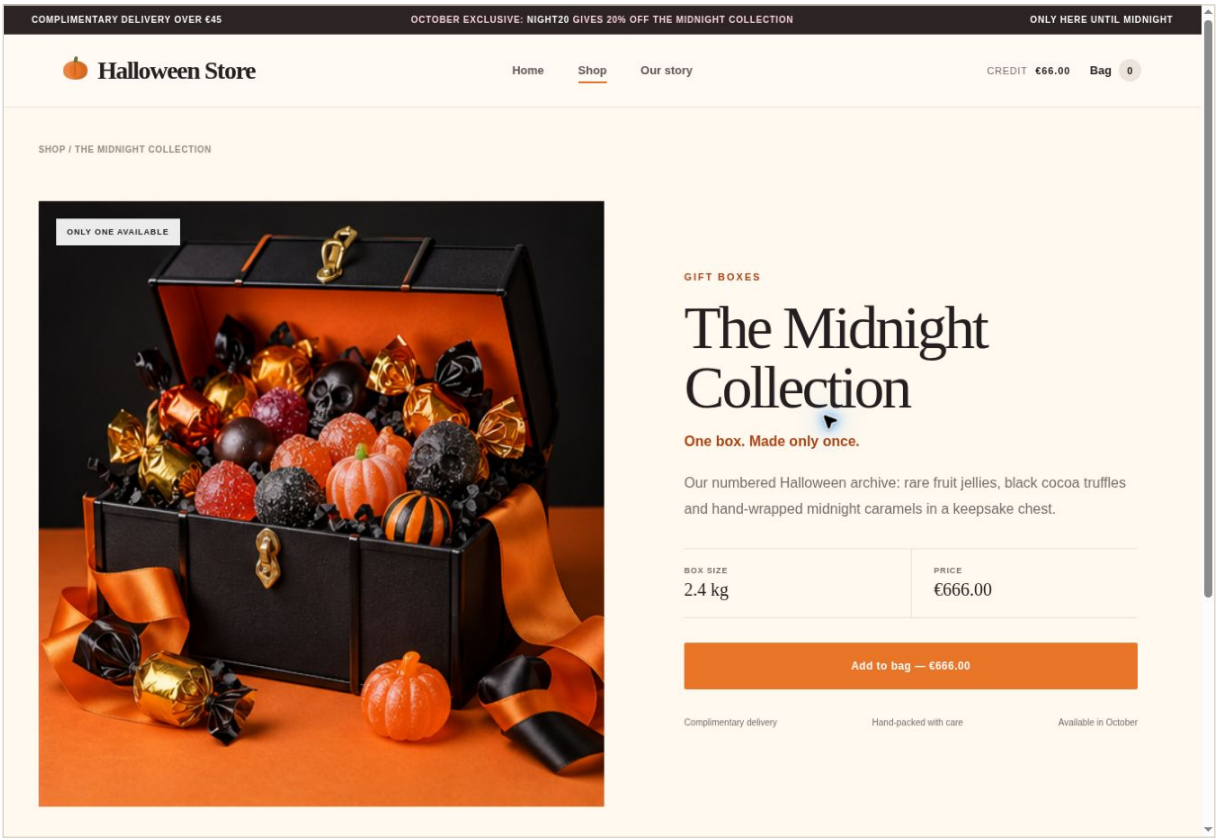



Figure 2. The genuine product page shows the target price and available credit. Route: /product/midnight-vault.

## The newsletter exposed a 30% promotion




Figure 3. Newsletter confirmation revealing NEWS30. Enlarged detail cropped from the original page screenshot.

The subscription was part of the challenge storefront. It required no external mailbox access. I used the returned code in the same browser session.

# Two valid codes still left €333 to pay

I added one Midnight Collection to the bag, then applied NEWS30 and NIGHT20 once each. Both discounts were calculated against the €666 subtotal, so their combined effect was 50%. Checkout remained disabled.

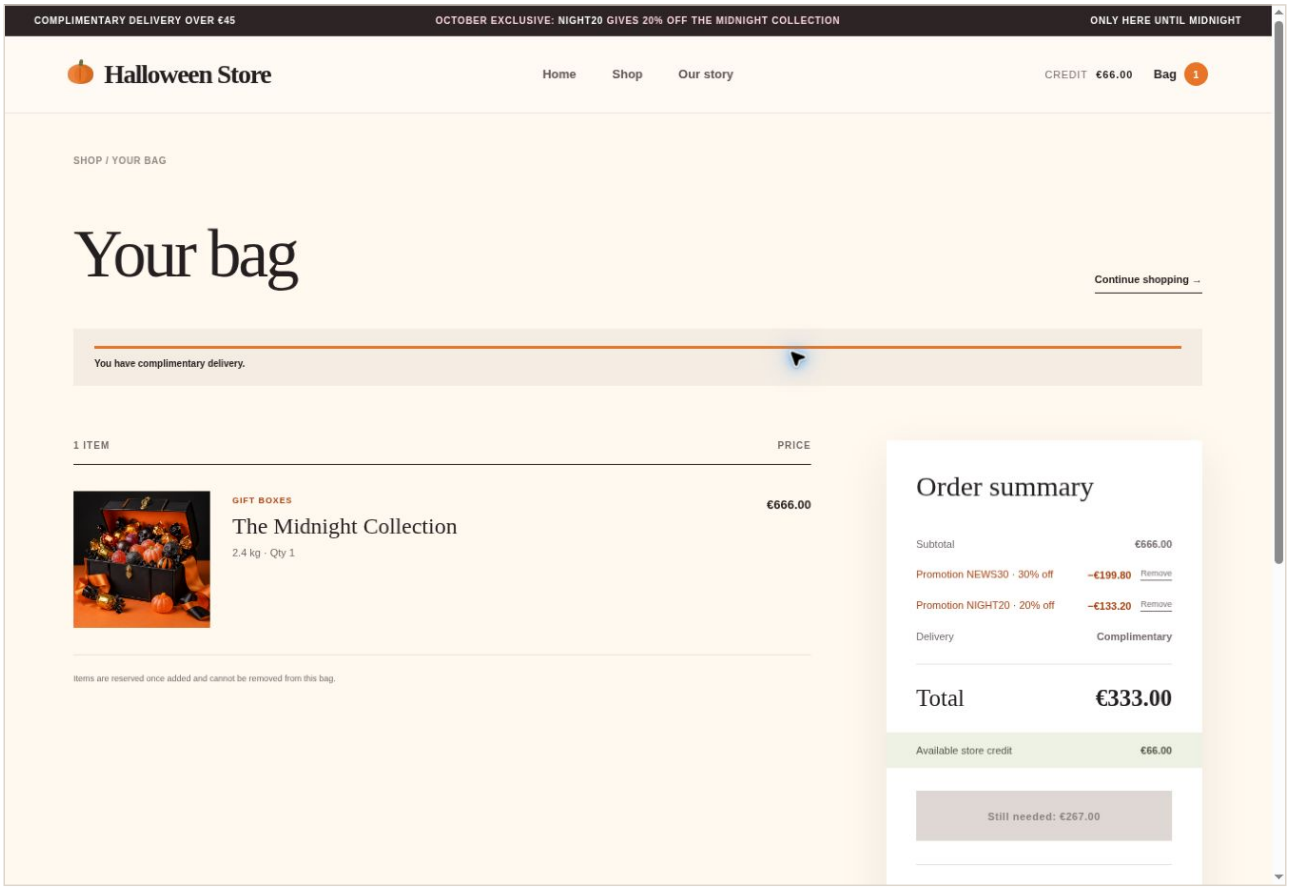



Figure 4. Both codes are active once; total €333.00, credit €66.00, and €267.00 still needed. Route: /cart.

| Component | Calculation | Amount |
|---|---|---|
| Subtotal | 1 × €666.00 | €666.00 |
| NEWS30 | 30% × €666.00 | -€199.80 |
| NIGHT20 | 20% × €666.00 | -€133.20 |
| Total | €666.00 - €333.00 | €333.00 |

This baseline mattered: ordinary coupon use did not satisfy the budget. The useful question was whether coupon uniqueness survived changes to the cart.

# NEWS30 accumulated instead of staying unique

After applying NEWS30 and then NIGHT20, I entered NEWS30 again. The live page accepted it and displayed NEWS30 × 2, with a €399.60 discount. The total fell to €133.20.

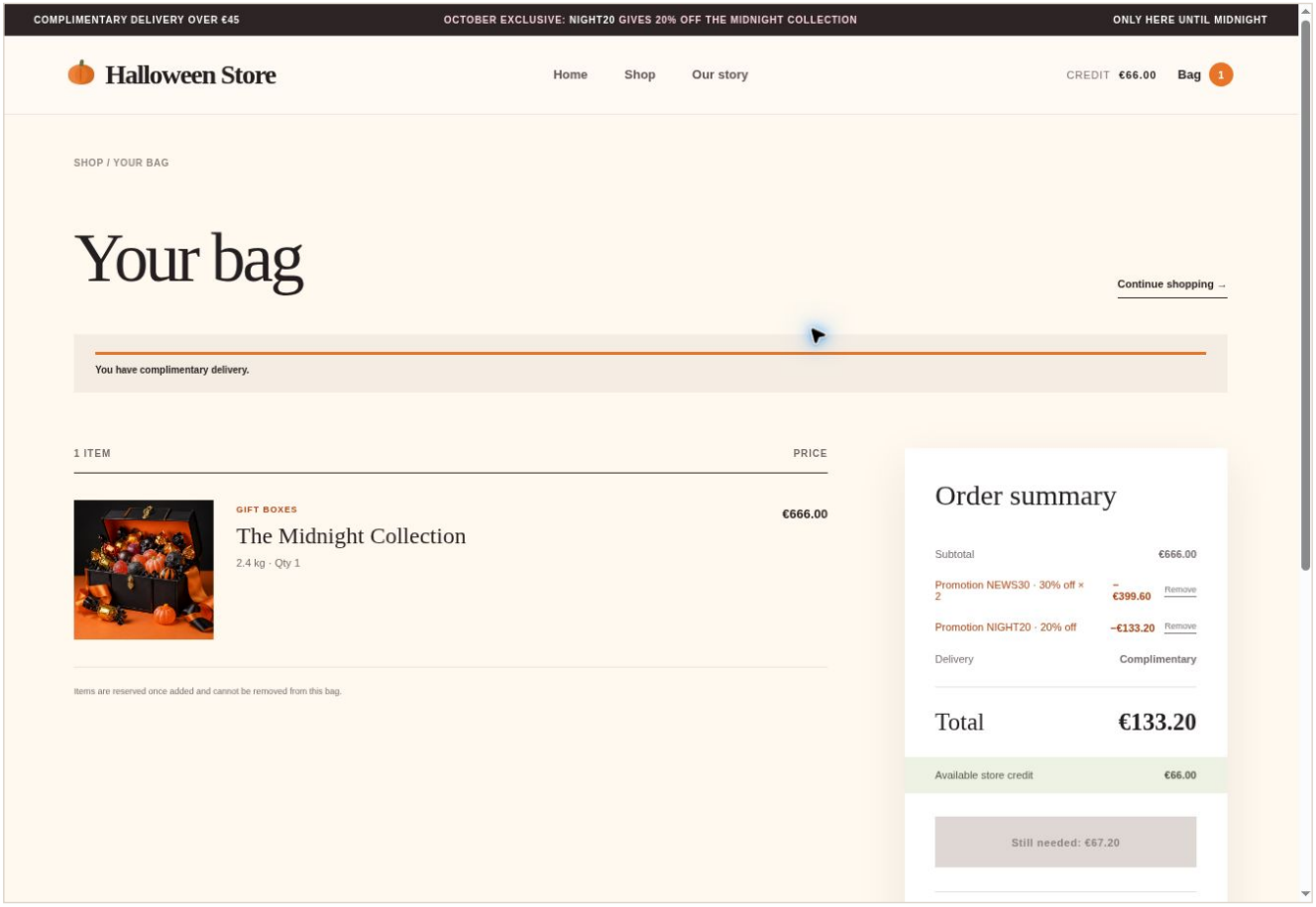



Figure 5. NEWS30 appears twice while NIGHT20 remains active. The order is still €67.20 above the available credit. Route: /cart.

> **A useful correction to the earlier draft**
> The first duplicate did not require a removal in this replay. The sequence NEWS30 → NIGHT20 → NEWS30 already produced × 2. Removing NIGHT20 was therefore not proven necessary for that first increment.

Control check: I immediately submitted NEWS30 once more. This time the page reported “Coupon already applied.” and kept the same total. That response was observed in the live interface; the transient message is not shown in Figure 5.

The evidence points to sequence-dependent duplicate handling. It does not establish an unlimited number of uses, a race condition, or the server’s internal data structures.

# I removed and restored the other coupon

With NEWS30 already at × 2, I removed NIGHT20. NEWS30 remained active and the total rose from €133.20 to €266.40, exactly the amount of the removed 20% discount.

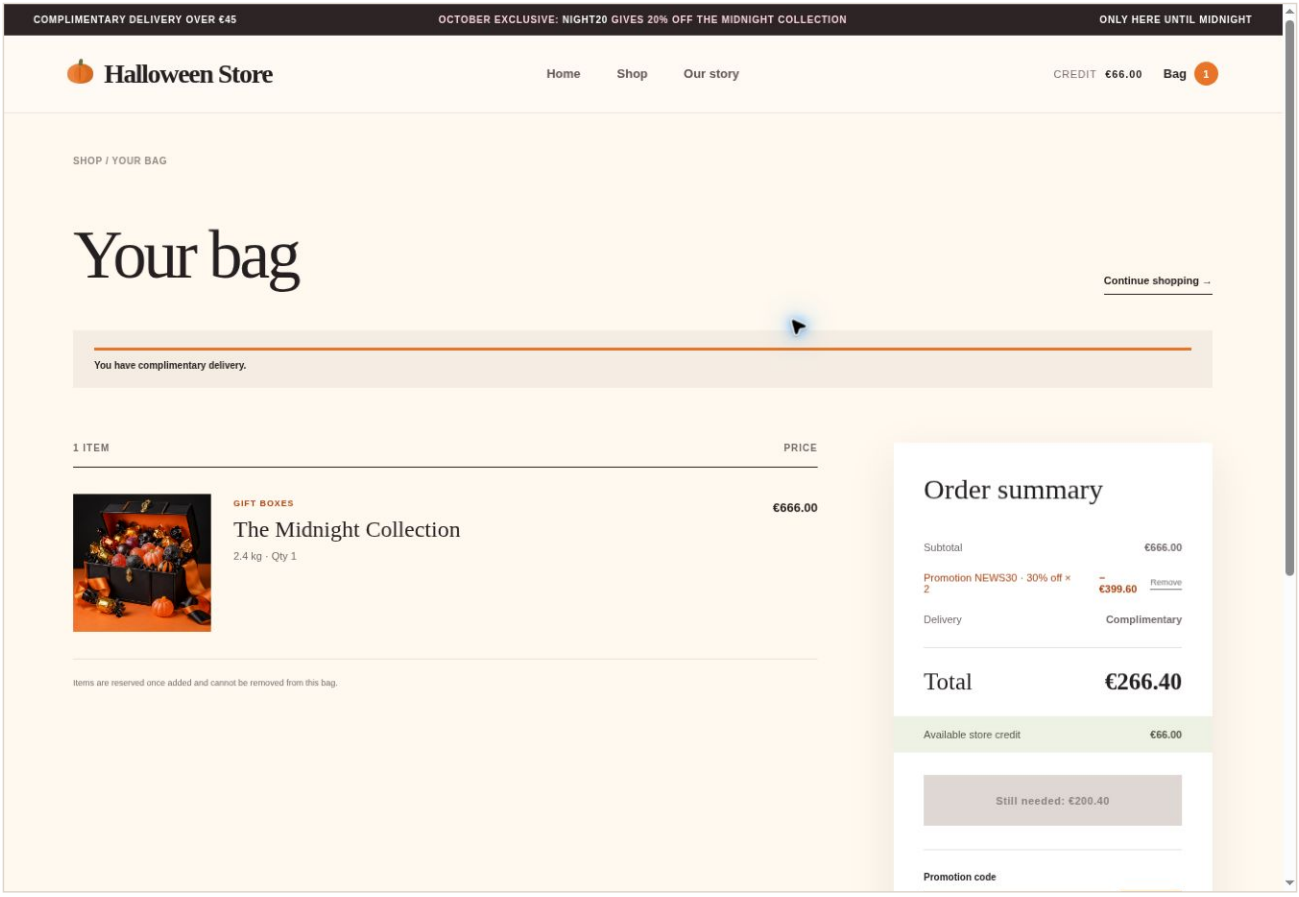



Figure 6. NIGHT20 is absent, NEWS30 × 2 is retained, and the total is €266.40. Route: /cart.

## I then completed the remove-and-reapply sequence

```
POST /api/cart/coupon/remove  {"code":"NIGHT20"}
POST /api/cart/coupon         {"code":"NIGHT20"}
POST /api/cart/coupon         {"code":"NEWS30"}
```

Restoring NIGHT20 returned the total to €133.20. Applying NEWS30 after that increased it to × 3. The endpoint mapping above comes from my original API proof of concept; this verification run exercised the same actions through the live form.

# The third NEWS30 application reached €0

The next NEWS30 application was accepted. The interface now showed a €599.40 NEWS30 discount and a €133.20 NIGHT20 discount. The total became €0.00, and the Complete order button became available.

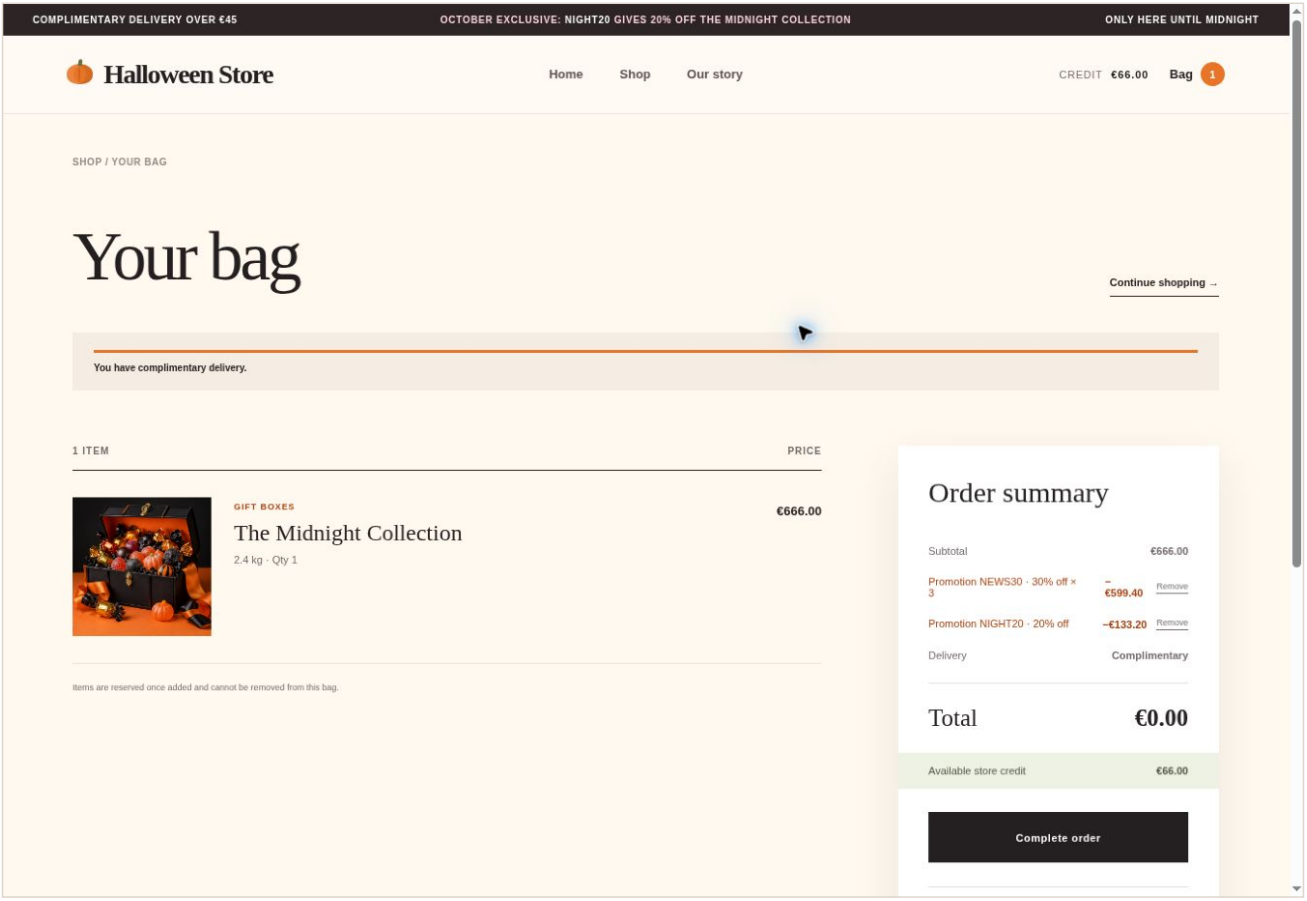



Figure 7. NEWS30 × 3, NIGHT20, a zero total, and an enabled Complete order button. Route: /cart.

| State | Raw discount | Payable total |
|---|---|---|
| NEWS30 × 1 + NIGHT20 | 50% = €333.00 | €333.00 |
| NEWS30 × 2 + NIGHT20 | 80% = €532.80 | €133.20 |
| NEWS30 × 3 + NIGHT20 | 110% = €732.60 | €0.00 |

The final displayed line-item discounts sum to €732.60, not €666.00. Their effective reduction of the purchase price is €666.00 because the payable total stops at zero. A zero floor prevents a negative total; it does not prevent an unauthorized free order.

# Checkout confirmed the result

I clicked Complete order. The challenge returned an order confirmation with €0.00 paid and the flag. The credit display remained €66.00. This confirms the result beyond a discounted cart preview.

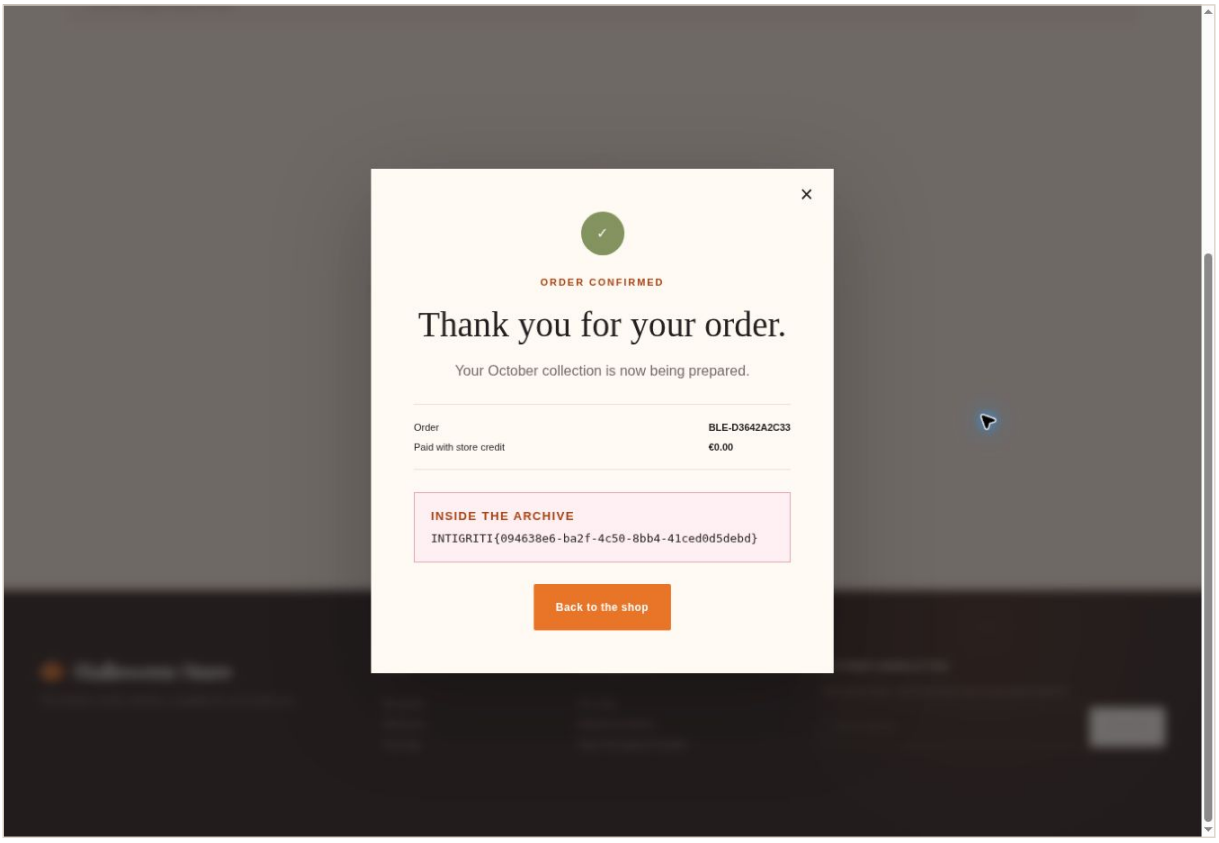



Figure 8. Genuine confirmation from the new replay: order BLE-D3642A2C33 and €0.00 paid. Route: /cart.

## Receipt detail




Enlarged crop of the same original screenshot, showing the order, paid amount, and flag.

**Flag:** INTIGRITI{094638e6-ba2f-4c50-8bb4-41ced0d5debd}

# What failed, and how I would fix it

The violated rule was simple: a single-use coupon should contribute at most one discount to a cart. The live interface accepted three NEWS30 applications, and checkout honored the resulting zero total.

| Supported by the replay | Not established by this evidence |
|---|---|
| Duplicate acceptance depends on action order. | A particular snapshot or object-reference implementation. |
| An immediate repeated NEWS30 was rejected. | A time-of-check/time-of-use race. |
| Changing NIGHT20 allowed the next NEWS30 increment. | That removal is always required, or stacking is unlimited. |
| The final order was paid at €0.00. | The impact on any production storefront. |

## Likely root cause

The behavior is consistent with inconsistent coupon-membership or deduplication state across cart transitions. The uniqueness check and accumulated discount state do not preserve the same invariant. Server source was not available, so the precise implementation remains a hypothesis.

## Recommended changes

**1. Enforce canonical uniqueness.** Store active coupons by normalized code and reject duplicates independently of action order.

**2. Make application idempotent.** Reapplying an active single-use code must leave its discount unchanged.

**3. Recompute discounts.** Derive totals from authoritative cart contents and currently eligible coupons after every mutation.

**4. Validate again at checkout.** Apply coupon limits and compatibility rules before accepting an order.

**5. Test state transitions.** Cover immediate duplicates, alternating codes, removal/reapplication, and checkout after each transition.

## Impact and scope

This was an intentionally vulnerable CTF shop. In a real store with the same flaw, unauthorized accumulated discounts could cause revenue loss. The demonstrated impact here is the successful €0 challenge order and returned flag.

# Reproduce the submitted API sequence

This bounded example adapts my original proof of concept, preserves one cookie session, and waits one second before each request. The live screenshots were captured through the browser; this revised PowerShell listing was not executed during that capture run.

```
$base = "https://challenge-1026.challenges.intigriti.io"
$s = New-Object Microsoft.PowerShell.Commands.WebRequestSession
function Req($method, $path, $body = $null) {
    Start-Sleep -Milliseconds 1000
    $requestParams = @{
        Method = $method; Uri = "$base$path"; WebSession = $s
        ErrorAction = "Stop"
    }
    if ($null -ne $body) {
        $requestParams.ContentType = "application/json"
        $requestParams.Body = $body | ConvertTo-Json -Compress
    }
    Invoke-RestMethod @requestParams
}
Req "GET" "/api/session" | Out-Null
Req "POST" "/api/newsletter" @{
    email = "researcher@example.com"
} | Out-Null
Req "POST" "/api/cart/items" @{
    productId = "midnight-vault"
} | Out-Null
Req "POST" "/api/cart/coupon" @{code="NEWS30"} | Out-Null
Req "POST" "/api/cart/coupon" @{code="NIGHT20"} | Out-Null
for ($round = 1; $round -le 2; $round++) {
    Req "POST" "/api/cart/coupon/remove" @{
        code="NIGHT20"
    } | Out-Null
    Req "POST" "/api/cart/coupon" @{code="NIGHT20"} | Out-Null
    $cart = Req "POST" "/api/cart/coupon" @{code="NEWS30"}
    $cart | ConvertTo-Json -Depth 12
}
Req "POST" "/api/checkout" @{} | ConvertTo-Json -Depth 12
```

## Evidence and disclosure

The eight figures use original live screenshots; the separate evidence archive also includes the undiscounted cart. Captures document a new replay, with a new order ID, and match the flag in my original submitted notes. No private Intigriti report was accessed or modified.

Official rules: automation limit 3 requests/second. Publish the public article only after 26 October 2026, 23:59 UTC (27 October, 04:59 Pakistan time), then add its link under the existing report. The contest requires a public webpage link; a PDF or raw Markdown file alone is not an entry.

Source: Intigriti Challenge 1026 official program rules
Live challenge: challenge-1026.challenges.intigriti.io/challenge