

espace-adherent

github.com/parti-rennaissance/espace-adherent

Scanned 29 July 2026 at 15:18 · Report generated 29 October 2026 · PHP · 779 stars

This repository is no longer available on GitHub. It was deleted or made private after this scan, so these findings cannot be re-verified. Read them as a record of what the repository contained on 29 July 2026.

OVERALL ASSESSMENT

High

Combined score 95 · 64 findings after filtering

Summary

The repository has a high vulnerability score of 95.0, which means there are many obvious coding errors that an attacker could easily exploit to break your system. However, the malicious-pattern score is 0.0, indicating that the code does not appear to be intentionally harmful or designed to attack others. Most of the issues found are standard mistakes like missing CSRF protection or SQL injection risks rather than signs of a hidden malware campaign. While some findings suggest potential for dangerous behavior, such as running containers as root or using user-controlled data in HTML, they are classified as insecure patterns that should be fixed to prevent real-world damage. The developer should prioritize addressing the high-severity issues first to ensure the application is safe and reliable before deploying it to production.

Written by a language model running locally on the scanning machine, from the findings in this report. The risk levels above and below were decided by the scoring model, not by this summary.

Risk, by type

MalDet scores two different questions separately, because they call for different responses: code can be badly written without being hostile, and hostile code is not always badly written.

VULNERABILITY

High (95)

Insecure coding an attacker could exploit. Usually mistakes, and usually fixable in place.

MALICIOUS PATTERN

Safe (0)

Signs the code may be trying to do harm: hidden payloads, credential access, persistence, exfiltration.

What each detector found

DETECTOR	LOOKS FOR	HIGH	MEDIUM	LOW	TOTAL
semgrep	Insecure coding patterns across many languages	14	50	0	64

7 of 64 findings were rated more likely genuine than noise by the classifier.

Evidence

Identical findings are grouped, so a single rule firing many times takes one row. Ordered worst first. “Seen” is how many times that finding occurred.

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
HIGH	semgrep	Generic Secret detected	2	/.env:21
HIGH	semgrep	By not specifying a USER, a program in the container may run as 'root'. This is a security hazard. If an attacker can control a process running as root, they may have control over the container. Ensure that the last USER in a Dockerfile is a USER other than 'root'.	2	/Dockerfile:92
HIGH	semgrep	User controlled data in methods like `innerHTML`, `outerHTML` or `document.write` is an anti-pattern that can lead to XSS vulnerabilities	10	/front/pages/assessor_manager_requests.js:24
MEDIUM	semgrep	CSRF protection is disabled for this configuration. This is a security risk. Make sure that it is safe or consider setting `csrf_protection` property to `true`.	3	/src/Controller/Renaissance/Adhesion/AdhesionController.php:57
MEDIUM	semgrep	RegExp() called with a `name` function argument, this might allow an attacker to cause a Regular Expression Denial-of-Service (ReDoS) within your application as RegExp blocks the main thread. For this reason, it is recommended to use hardcoded regexes instead. If your regex is run on user-controlled	1	/front/utils/url.js:2

SEVERITY	DETECTOR	FINDING	SEEN	EXAMPLE LOCATION
MEDIUM	semgrep	Detected string concatenation with a non-literal variable in a Doctrine DBAL query method. This could lead to SQL injection if the variable is user-controlled and not properly sanitized. In order to prevent SQL injection, use parameterized queries or prepared statements instead.	2	/src/Command/SyncEventInscriptionDuplicatesCommand.php:56
MEDIUM	semgrep	Depending on the context, user control data in `Object.assign` can cause web response to include data that it should not have or can lead to a mass assignment vulnerability.	1	/front/listeners/em-datetime-picker.js:12
MEDIUM	semgrep	The `redirect()` method does not check its destination in any way. If you redirect to a URL provided by end-users, your application may be open to the unvalidated redirects security vulnerability. Consider using literal values or an allowlist to validate URLs.	40	/src/Controller/Admin/AdminCandidatesGroupCandidateImportController.php:94
MEDIUM	semgrep	Detection of dangerouslySetInnerHTML from non-constant definition. This can inadvertently expose users to cross-site scripting (XSS) attacks if this comes from user-provided input. If you have to use dangerouslySetInnerHTML, consider using a sanitization library such as DOMPurify to sanitize your HT	2	/front/components/DataGrid.js:318
MEDIUM	semgrep	This tag is missing an 'integrity' subresource integrity attribute. The 'integrity' attribute allows for the browser to verify that externally hosted files (for example from a CDN) are delivered without unexpected manipulation. Without this attribute, if an attacker can modify the externally hosted	1	/public/maintenance.html:27

How to read this report

- **This is a static scan.** The code was read, never executed. Behaviour that only appears at runtime is outside what any of these detectors can see.
- **Findings are not all equally real.** A classifier rates each finding on how likely it is to be genuine rather than noise, and that rating weights the scores. Measured against a hand-reviewed sample it reached precision 0.52 and recall 0.62 — useful for ranking, not a verdict on any single finding.

- **A high score is not proof of malice.** Security tools, penetration-testing frameworks and malware research repositories contain the same patterns as real malware by design. The two risk types above exist to keep that distinction visible.
- **Absence of findings is not proof of safety.** A detector that is not installed, or a language none of them parse, produces silence rather than a warning.
- **This scan is a snapshot.** The repository may have changed since 29 July 2026.