# Security Vulnerability Report

Application: Password Strength Checker

Tools Used: Bandit (SAST), Safety (SCA), OWASP ZAP (DAST)

Date: October 2023

## 1. Bandit (Static Analysis)

*Bandit scans python source code for common security issues.*

### Findings

* **Issue:** B101: assert\_used
  + **Severity:** Low
  + **Description:** Use of assert detected in tests/test\_basic.py.
  + **Resolution:** Marked as False Positive (Intended). Assert is standard for unit testing frameworks like Pytest. It is not present in production src/ code.
* **Issue:** B201: flask\_debug\_true
  + **Severity:** High
  + **Description:** app.run(debug=True) found in run.py.
  + **Resolution:** Modified run.py to only use debug mode if explicitly set, or ensure run.py is not used in production deployment (WSGI server usage).

## 2. Safety (Dependency Check)

*Safety checks installed PyPI packages against known CVE databases.*

### Findings

* **Status:** **CLEAN**
* **Details:** Scanned flask, flask-login, flask-bcrypt, pyotp. No known vulnerabilities were found in the installed versions (Oct 2023).

## 3. OWASP ZAP (Dynamic Analysis)

*ZAP attacks the running application to find runtime flaws.*

### Findings

* **Issue:** Missing Anti-Clickjacking Header
  + **Severity:** Medium
  + **Description:** The response does not include X-Frame-Options or Content-Security-Policy.
  + **Remediation:** Added middleware to inject X-Frame-Options: SAMEORIGIN to all responses.
* **Issue:** Cookie No HttpOnly Flag
  + **Severity:** Low
  + **Description:** Session cookie set without HttpOnly flag.
  + **Remediation:** Configured SESSION\_COOKIE\_HTTPONLY = True in config.py.

## 4. Conclusion

The application has passed basic security linting. Critical findings regarding debug modes were remediated. Future work should focus on HTTP Security Headers hardening.