Finding 3 — CSRF (Cross-Site Request Forgery) — Weak Token Validation

Location / Page: [state-changing endpoint used in PoC — e.g., profile/settings POST]

Observation: Token validation logic appears to accept any request that *contains a token* without properly validating its authenticity (i.e., the server checks token presence but not correctness). This creates a weak CSRF protection where an attacker can supply a token value and the server will accept it.

Steps to reproduce:

1. Host the PoC HTML on an attacker-controlled page.
2. While victim is authenticated and visits attacker page, auto-submit or click the form.
3. Because the server requires only the presence of a token field (not validation), the action is performed.

Impact (Severity: Medium/High):
Unauthorized state-changing actions can be performed on behalf of authenticated users (change profile, alter settings, etc.). Weak token validation negates the protection expected from CSRF tokens.

Recommendation / Mitigation:

- Generate unpredictable CSRF tokens server-side, bind them to the session, and validate token value on each request.
- Do not consider mere presence of a token sufficient — verify the token matches the server-stored/session token.
- Use `SameSite` cookie attribute, validate `Origin`/`Referer` headers on sensitive endpoints, and require re-auth for high-risk actions.