Security Testing — Worksheet & Test Plan

# Activity A — Input Validation Test Table (pairs)

Fill the table below for each field you test. Execute two live tests if a safe lab page is available, or reason through expected results.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Test input | Expected behaviour | Actual behaviour | Pass/Fail | Notes |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# Activity B — Authentication & Authorisation Checks (pairs)

Use this checklist to test authentication and authorisation in simple web app. Record evidence (screenshot or network request).

* Authentication checks
* Attempt multiple failed logins (e.g., 6 attempts). Expected: account lockout or rate limiting.
* Test password policy: try weak password on registration (e.g., 'password'). Expected: rejected or prompted for stronger.
* Check remember-me behavior and cookie flags (HttpOnly, Secure, SameSite). Use DevTools → Application → Cookies.
* Authorisation checks
* Forced browsing: while logged in as a normal user, request admin URLs (e.g., /admin). Expected: 403 or redirect to login.
* IDOR check: change object id in URL (e.g., /orders/2 → /orders/3). Expected: 403 or 404 if you don't own it.
* Role tampering: check client-stored role flags (localStorage/session). Server must enforce role checks; tampering client-side should not grant access.

# Activity C — Short Security Test Plan (individual)

Use this template to write a one-page test plan for a simple web form or small app. Keep it concise — 1 page maximum.

Title:

Scope: (what part of the app you will test)

Tools: (e.g., Browser DevTools, Postman, screenshots)

Timebox: 20–30 minutes

Checks (for each, describe how to test and expected secure behaviour):

1) Authentication:
 - How to test:
 - Expected result:

2) Input validation:
 - How to test:
 - Expected result:

3) Authorisation:
 - How to test:
 - Expected result:

4) Sessions & cookies:
 - How to test:
 - Expected result:

**Deliverable (what you need to complete)**

* Completed input test table (one per pair).
* One-page test plan (individual).
* One evidence screenshot or saved Network request file showing a key test.

# Homework

Write a 300-word summary: pick one check from your test plan, describe a likely failure mode, and give three developer recommendations to fix it (e.g., parameterised queries, input sanitisation, server-side role checks).

## Evidence examples (simulated)

Below are three simulated screenshots you can use as examples to guide your evidence capture. These are safe, anonymised simulations for classroom use. When you capture real evidence, include the Network request body and a timestamped server log entry if available.



Figure 1: Tautology login bypass (simulated) — capture Network request body showing injected input.



Figure 2: Oversized input / boundary test (simulated) — capture the request/response showing truncation or error.



Figure 3: Cookie flags inspection (simulated) — capture HttpOnly, Secure and SameSite flags in DevTools.

## How to capture a Network request in DevTools (step-by-step)

1. Open Chrome and press F12 (or right-click → Inspect) to open DevTools.
2. Click the 'Network' tab.
3. Ensure 'Preserve log' is checked so the request remains visible.
4. Perform the action (e.g., submit the login form).
5. Click the POST request (e.g., /rest/user/login) and view the 'Request Payload' or 'Form Data' panel.
6. Right-click the request and choose 'Save as HAR with content' or take a screenshot of the Request/Response panels.
7. Save server logs (if you control the server) by tailing them during the test (e.g., `docker logs -f juice-shop` or `pm2 logs`).

## Assessment rubric (reminder)

Total 20 marks:
 - Evidence & reproduction (6 marks): clear screenshots, network request captured (0–6)
 - Correct classification of severity (6 marks): appropriate High/Medium/Low (0–6)
 - Quality of remediation suggestions (6 marks): correct and practical (0–6)
 - Presentation & clarity (2 marks): concise and readable (0–2)