Great, this is **WSTG-SESS-07: Testing for Logout Functionality & Session Timeout**.  
Here’s the structured breakdown 👇

**🧪 Testing for Logout Functionality & Session Timeout**

**🔎 Why Important?**

* **Logout functionality** ensures a user can terminate their session **immediately**.
* **Session timeout** ensures inactive sessions are killed to reduce risks of session hijacking (e.g., unattended computers, stolen cookies).
* Weak implementation = attacker can **reuse stolen cookies** or keep access after logout.

**🔧 How to Test Logout Functionality**

1. **Check if Logout Truly Invalidates Session**
   * Log in → capture the session cookie (via **Burp Proxy**).
   * Logout from the application.
   * Reuse the old cookie to access a protected page.
     + ❌ If it still works → **logout is broken** (session not invalidated server-side).
2. **Check for Client-Side Logout Only**
   * Some apps just delete the cookie in the browser with JavaScript.
   * If session still valid on server → attacker with stolen cookie can continue.
3. **Multi-Session Behavior**
   * Login in **two browsers/devices**.
   * Logout in one → see if other session is still valid.
   * If yes → only local logout, not server-side.

**🔧 How to Test Session Timeout**

1. **Idle Timeout**
   * Stay idle for X minutes.
   * Try accessing the app again.
   * If still active after long idle period → ❌ weak timeout.
2. **Absolute Timeout**
   * Stay logged in for several hours (even if active).
   * Check if session expires after maximum allowed time (e.g., 8 hours).
3. **Cookie Expiry**
   * Inspect cookies in browser dev tools.
   * If **expiry = far future (days/weeks)** → insecure.
   * Secure apps use **short-lived cookies** + refresh with re-authentication.
4. **Concurrent Session Reuse**
   * Save cookie.
   * Logout.
   * Reload saved cookie.
   * If it works → ❌ session not invalidated properly.

**✅ Best Practices / Mitigations**

* On logout:
  + **Destroy server-side session**.
  + **Invalidate session tokens**.
  + Redirect user to login page.
* For timeouts:
  + **Idle timeout**: 10–15 minutes for sensitive apps (e.g., banking).
  + **Absolute timeout**: enforce logout after e.g., 8–12 hours.
* Use **short-lived tokens** with refresh mechanism.
* Re-authenticate for **critical actions**.
* Combine with **HttpOnly, Secure, SameSite** cookie flags.

⚡ **Summary:**

* Test logout: does the session token still work after logout?
* Test timeout: do idle/absolute timeouts exist & are enforced?
* Strong apps → server-side invalidation + strict timeout policies.

👉 Do you want me to also write a **Burp Suite lab-style step-by-step workflow** for testing logout & session timeout (like how we did for CSRF & session fixation)?