**Challenge Descriptions**

**1. SQL Injection (Login Bypass)**

**Description:**  
A login form is vulnerable to SQL injection. User input is directly inserted into the SQL query without sanitization.

**Goal:**  
Bypass authentication and gain access to the admin panel.

**Optional Hint:**

* Try: ' OR '1'='1 --
* Inspect the login request and test payloads.

**2. Cross-Site Scripting (XSS — Stored)**

**Description:**  
A feedback form stores user input without sanitizing it. This can lead to malicious JavaScript execution when viewed by an admin.

**Goal:**  
Trigger your script when the admin checks your feedback.

**Optional Hint:**

* Test with: <script>alert('XSS')</script>
* Remember: Stored XSS triggers when another user views it.

**3. Command Injection**

**Description:**  
The diagnostics page uses your input in a shell command (ping) without sanitization, allowing command chaining.

**Goal:**  
Execute arbitrary system commands.

**Optional Hint:**

* Append ; ls to your input.
* Test if && or | works.

**4. Insecure File Upload**

**Description:**  
The upload form does not validate file types or prevent execution of uploaded files.

**Goal:**  
Upload a malicious script and execute it.

**Optional Hint:**

* Rename .php to .jpg but keep PHP code.
* Access the uploaded file directly in the browser.

**Troubleshooting**

* **Port Already in Use**

bash

CopyEdit

docker ps

docker stop <container\_id>

* **Container Exits Immediately**

bash

CopyEdit

docker logs <container\_id>

* **Image Build Fails**  
  Check your internet connection and make sure Docker is running.