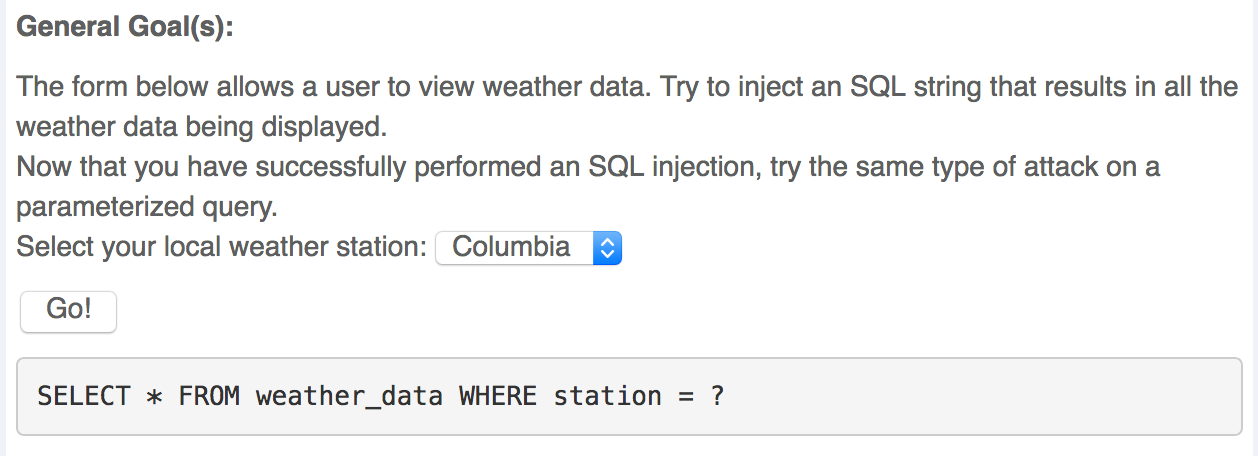
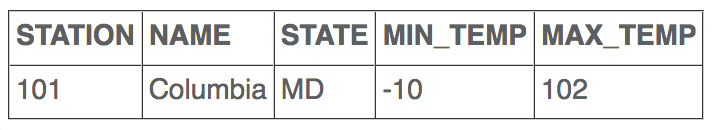
網路與多媒體實驗 hw2 Webgoat 電機四B03901153 陳楷訓

1. Numeric SQL Injection :

題目截圖：



解題流程：

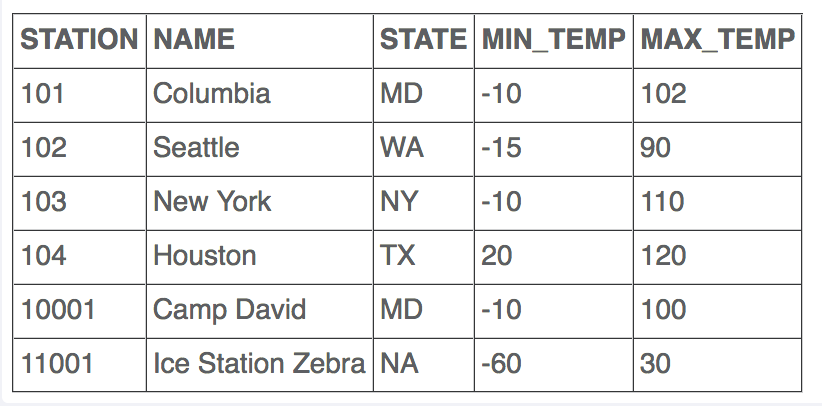


SELECT \* FROM weather\_data WHERE station = 101

想要列出所有weather data

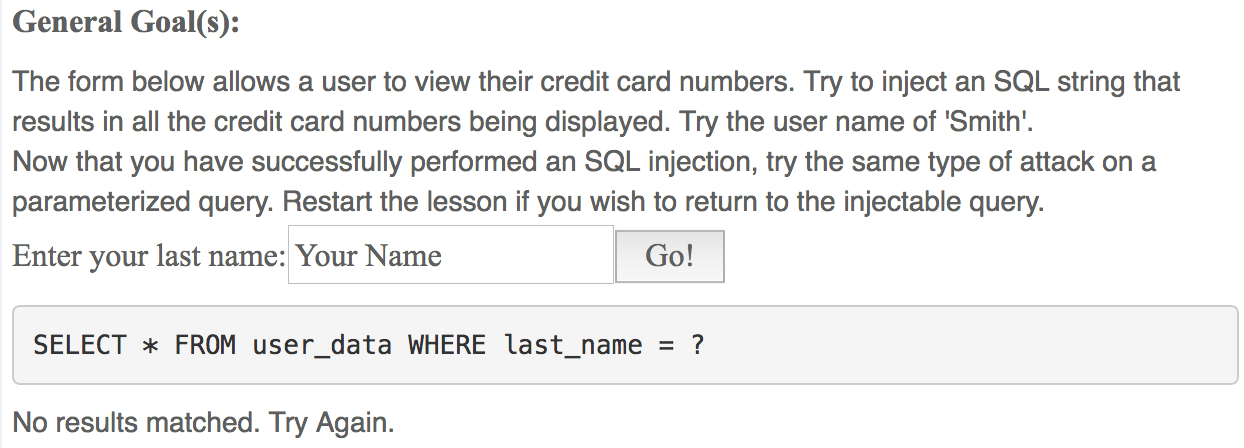
🡪 使用burp攔截封包，將station = 101，改成 station = 101 or 1=1

結果。



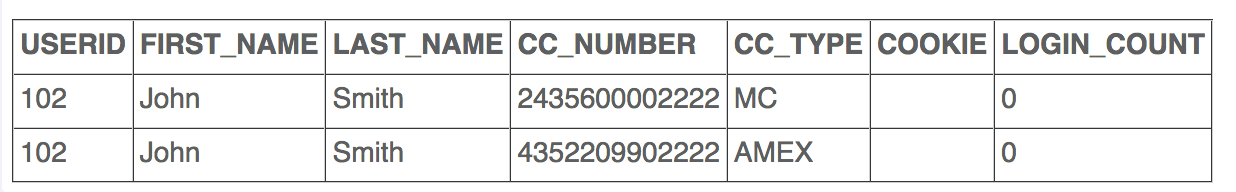
1. String SQL Injection :

題目截圖：



解題流程：

先在空格中輸入 Smith



思考由於Smith為一個string所以SQL指令會是：

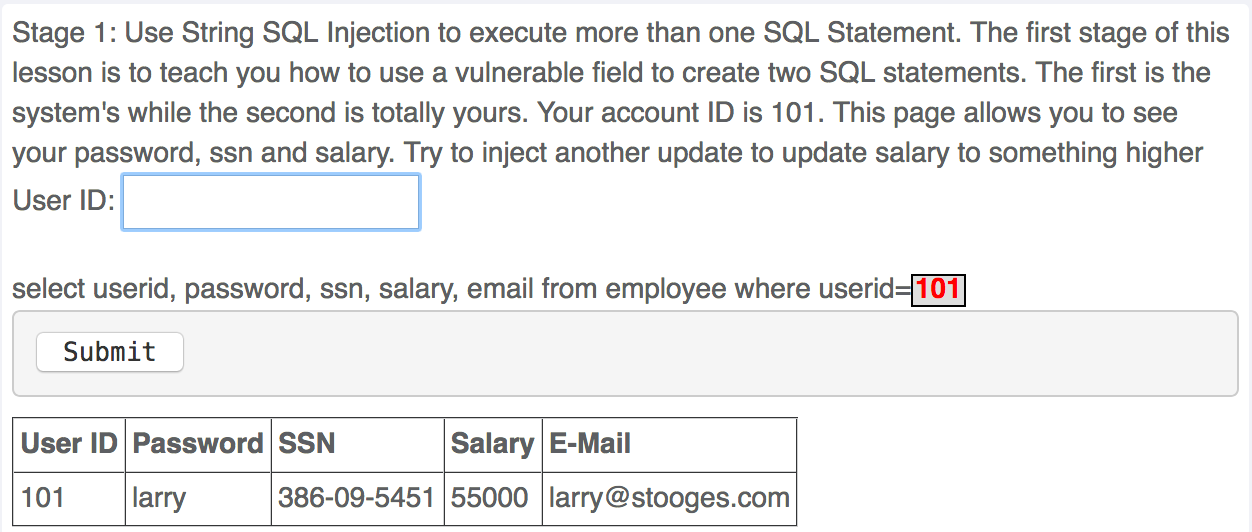
SELECT \* FROM user\_data WHERE last\_name = ‘Smith’

所以若是攔截封包並且改成Smith’or 1=1 /\* 則SQL指令會變成

SELECT \* FROM user\_data WHERE last\_name = ‘Smith’or 1=1 /\*’。

1. Database backdoor : Stage1

題目截圖：



解題流程：

猜測SQL指令為：

select \* from employee where userid = ?

所以若在User ID 中輸入：

101; update employee set salary=66666 where userid = 101

則SQL指令將變為：

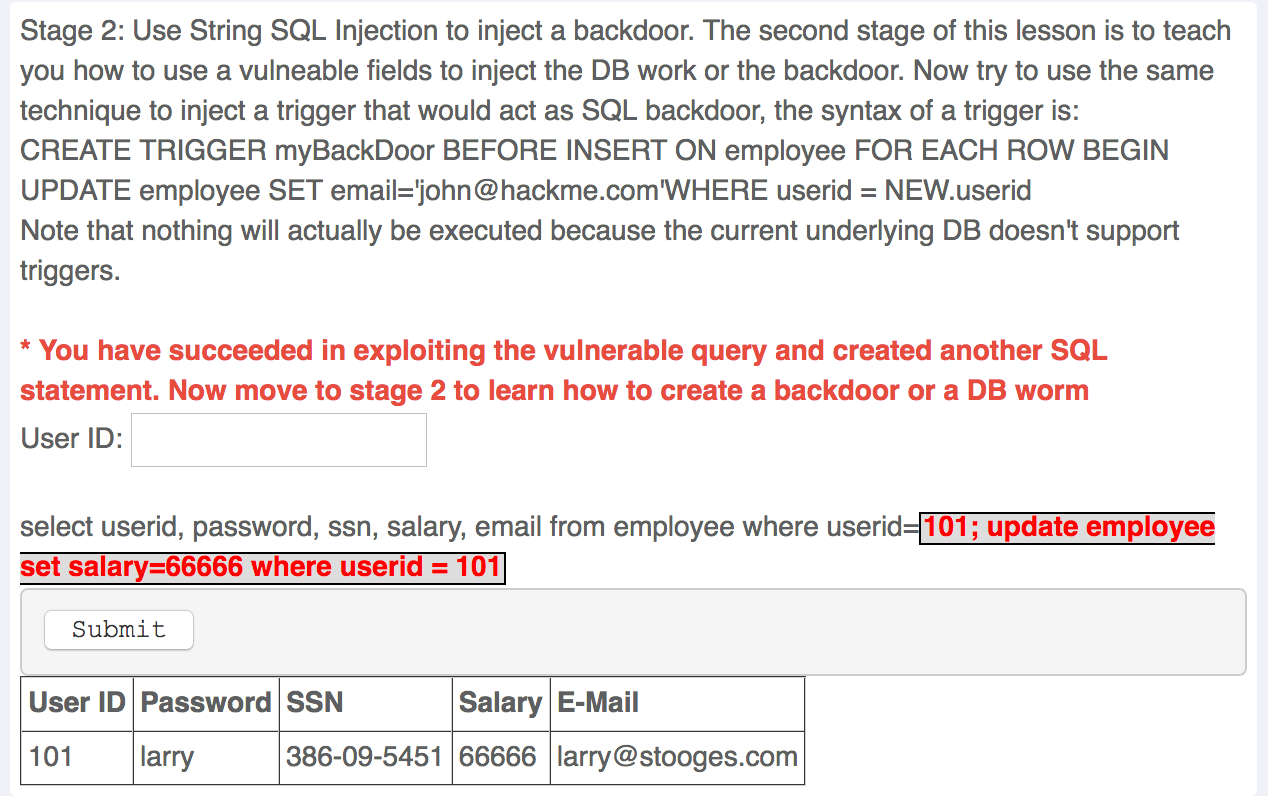
select \* from employee where userid = 101; update employee set salary=66666 where userid = 101

即可將userid為101的employee的salary改成66666。



1. Database backdoor : Stage2

題目截圖：



解題流程：

因為題目說 use the same technique，所以我想也是加分號使空格中不只含有一個SQL statement。

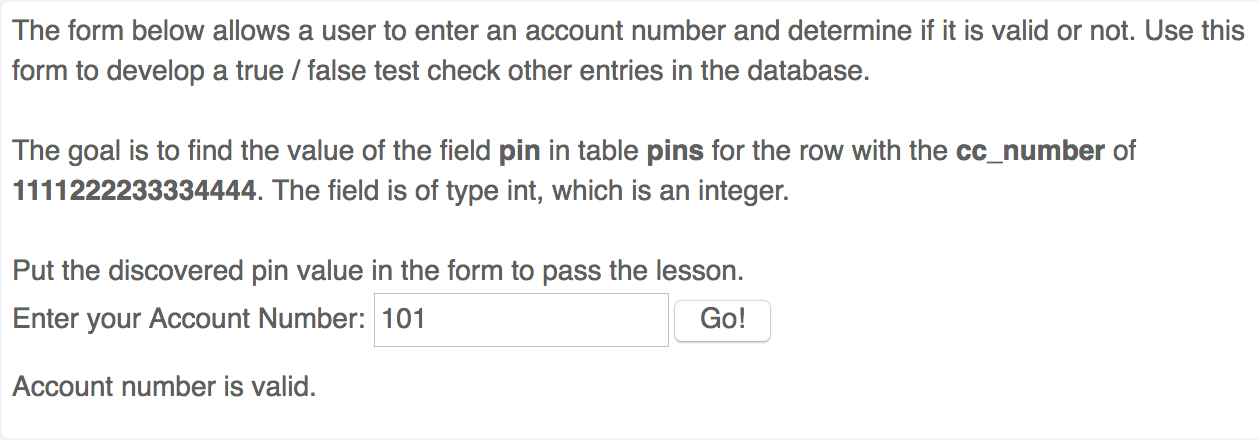
在空格中輸入：

101; CREATE TRIGGER myBackDoor BEFORE INSERT ON employee FOR EACH ROW BEGIN UPDATE employee SET email='john@hackme.com'WHERE userid = NEW.userid

即通過此關，而題目敘述有說由於此db不support trigger所以沒有成果截圖。

1. Blind Numeric SQL Injection

題目截圖：



解題流程：

猜測後端function類似：

if( isValid(Account\_Number) ){

print(“Account is valid”)

}else{

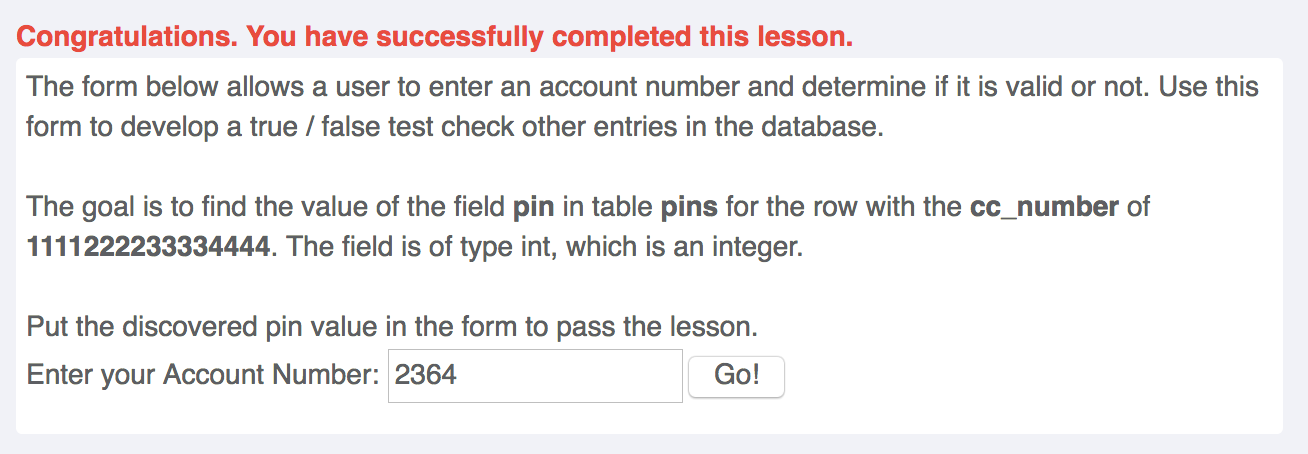
print(“Invalid account number”)

}

若要測試pin，則使用 101 AND **((SELECT pin FROM pins WHERE cc\_number='1111222233334444') > 1000 );**

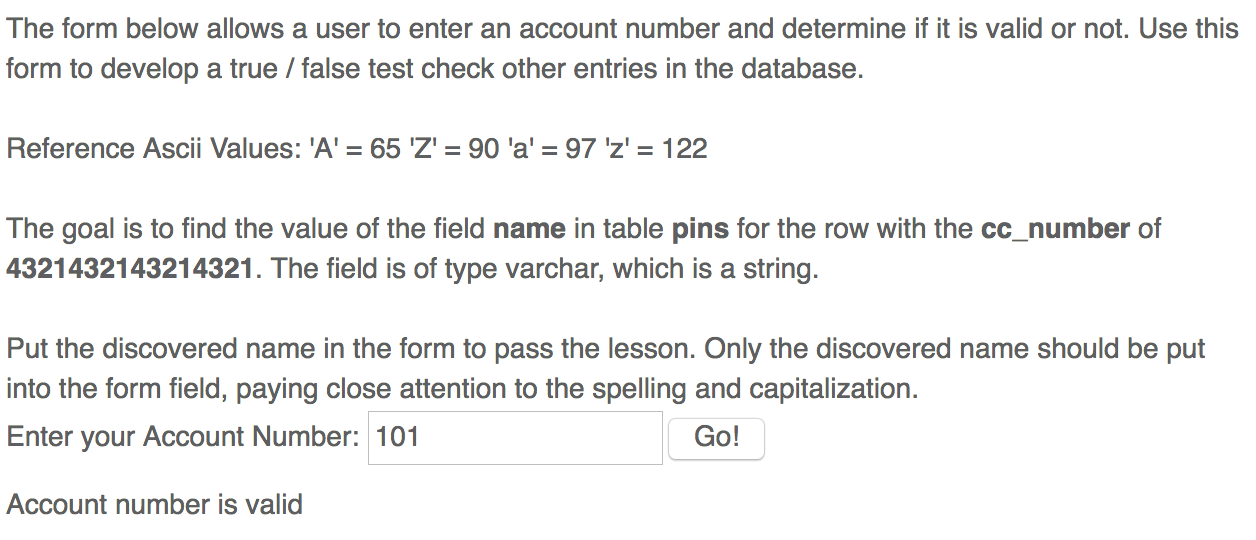
等效於 True AND (boolean)，因為已知Account\_Number = 101是True

慢慢迭代縮小範圍得，pin = 2364。



1. Blind String SQL Injection

題目截圖：



解題流程：

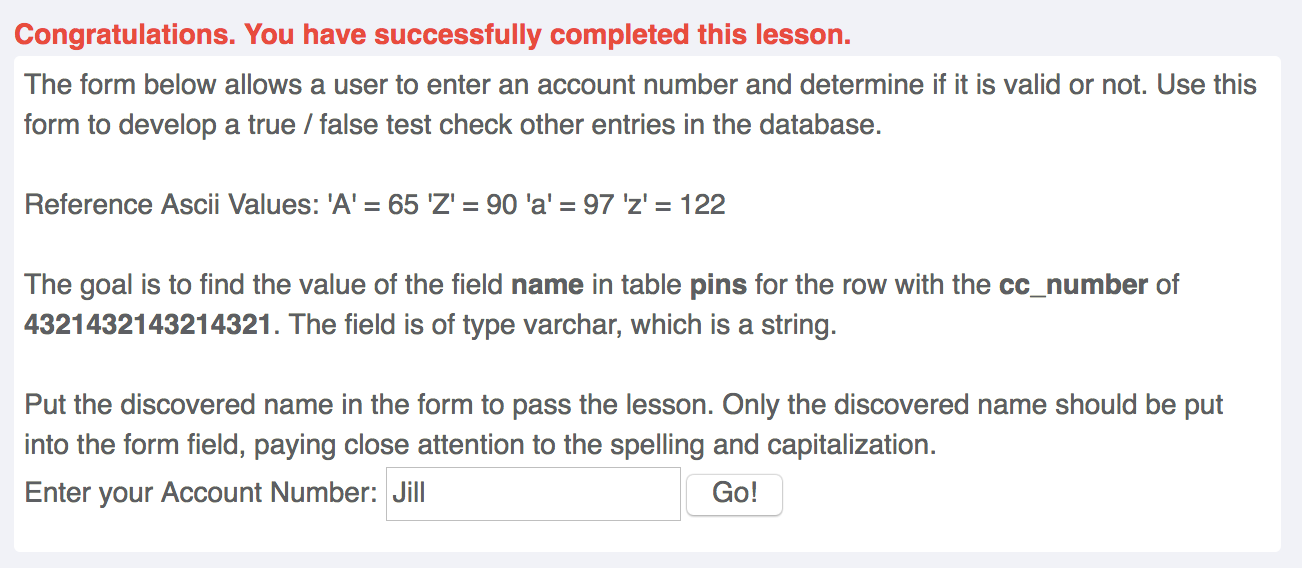
一開始猜想指令會是長這樣

101 AND ((SELECT name FROM pins WHERE cc\_number='4321432143214321') = 'Kevin' )

但嘗試了幾個常見的名字以後果斷放棄，覺得這樣嘗試究竟何年何月可以完成，於是我再重新看一次題目，發現題目有提及ASCII value，我推測應該會是取出一個個char比較，得出一個個char最後組成name。

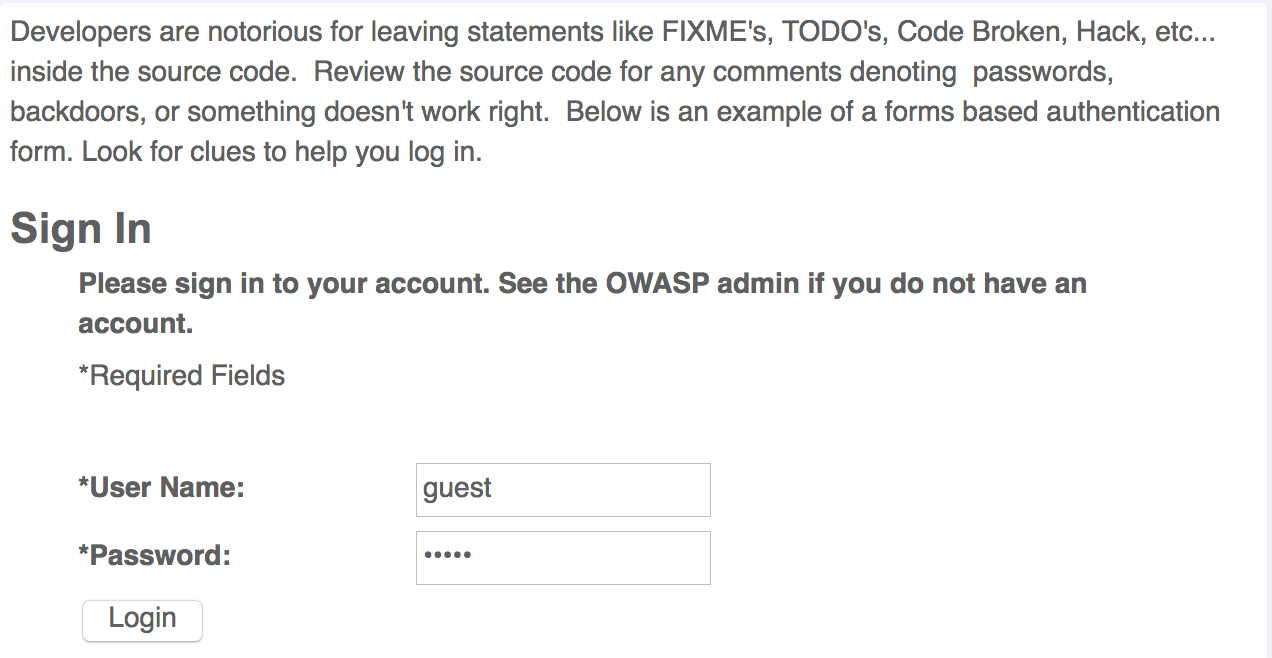
101 AND (SUBSTRING((SELECT name FROM pins WHERE cc\_number='4321432143214321'), 1, 1) = 'J' );

Ans : Jill



1. Discover Clues in the HTML :

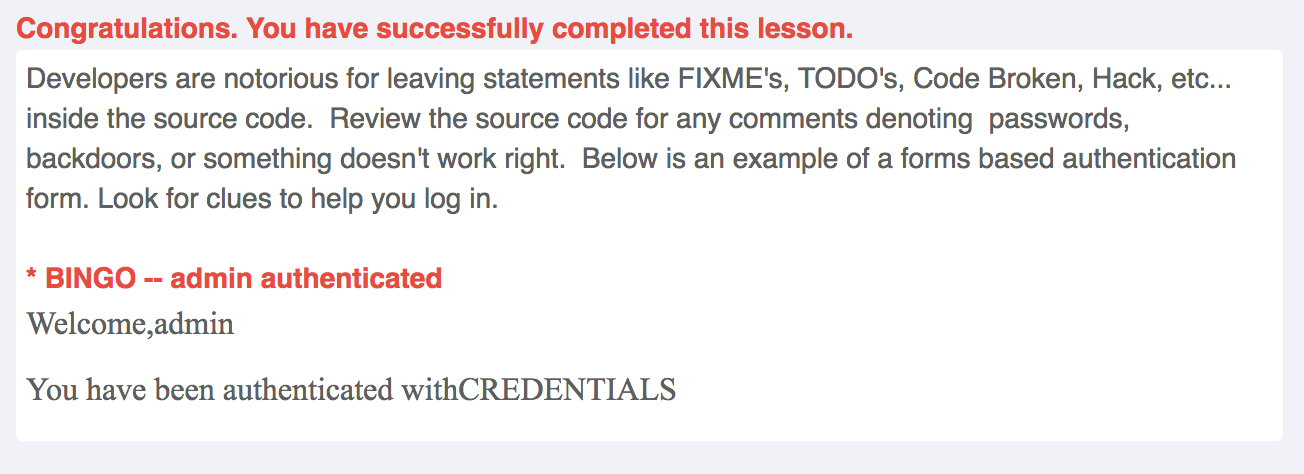
題目截圖：



解題流程：

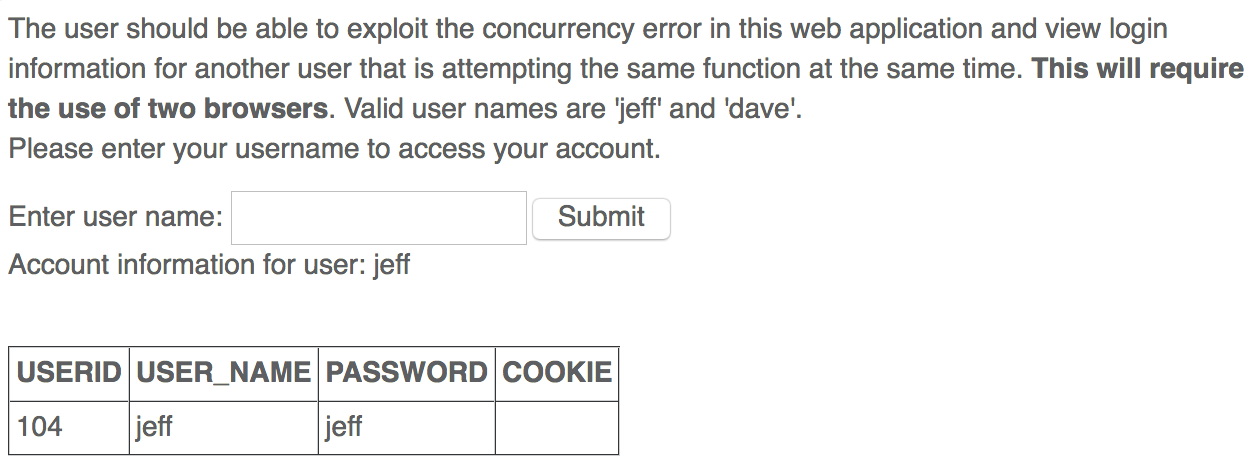
在firefox中點擊「檢測元素」，看到<!--FIXME admin:adminpw-->

於事輸入User Name : admin , Password : adminpw，成功登入。



1. Thread Safety Problems

題目截圖：



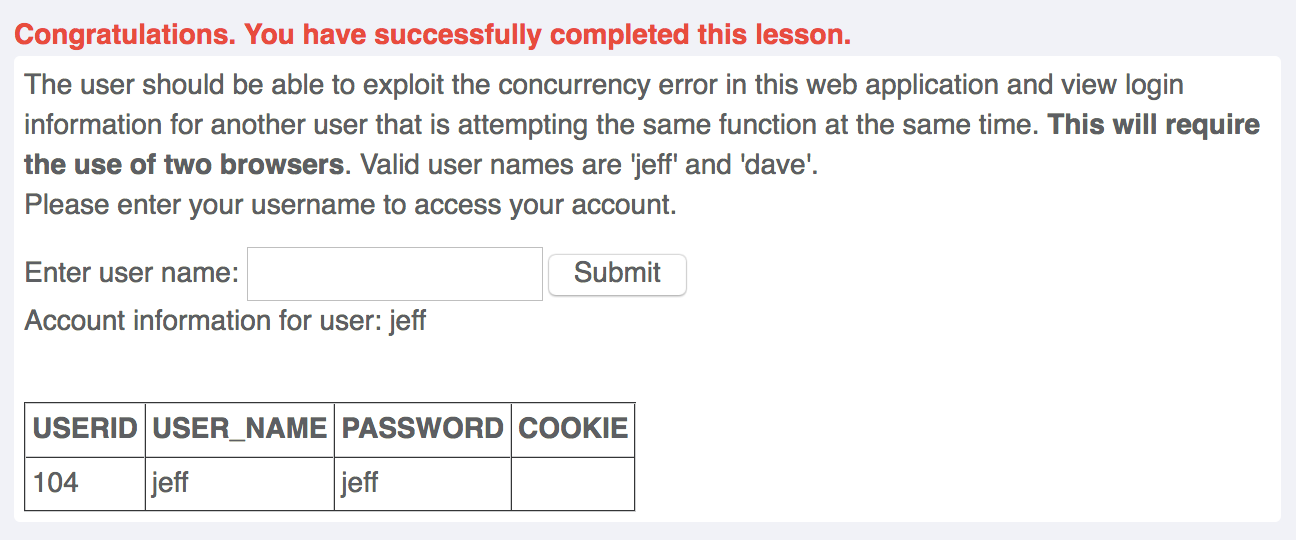
解題流程：

由於我不了解Thread Safety 的原理於是我先去google一下。

「但我們知道static變數，在程式中如果要操作，應該要進行lock來避免有其他thread同時存取這個變數，但如果你沒做，你就會發現在使用人數較多的web系統上，異常的錯誤就會出現，而且很容易引導你朝向錯誤的方向去找這個問題，你可能會以為是多人連線造成系統錯誤，又或者其他怪問題(EX:兩個Thread相繼讀寫同一個static變數，讀出來的結果卻是對方設定的值，整個錯亂)，但其實這個錯誤都是因為Thread Safe的issue了」

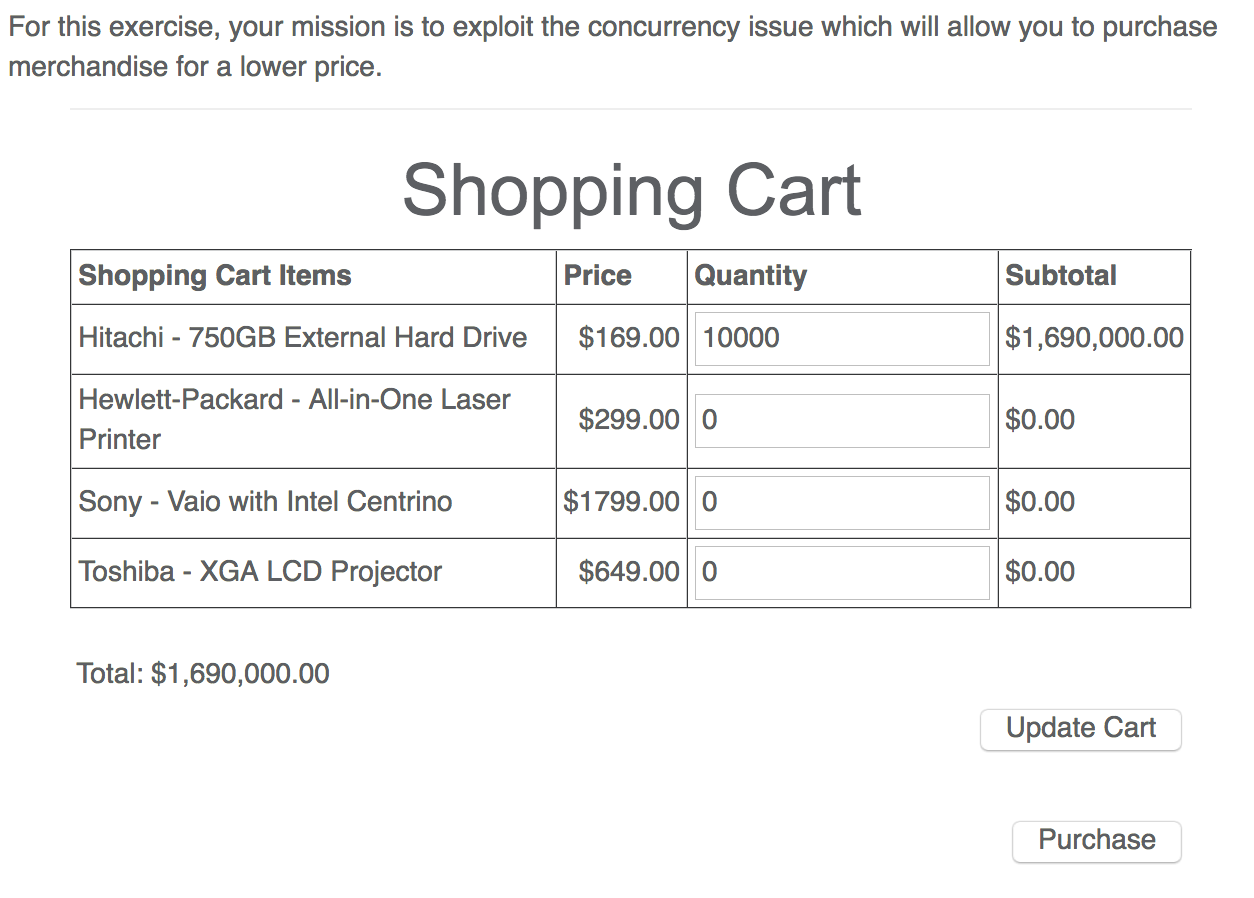
總之就是在多人連線時，若同時對同一個static variable進行操作，則可能會產生thread safety problem。

所以我必須要製造「多人連線」的情況誘發thread safety problem產生，於是我開啟兩個分頁，並且分別輸入 jeff 和 dave並且快速submit，試了幾次以後成功使兩個分頁都印出jeff的資料。



1. Shopping Cart Concurrency Flaw

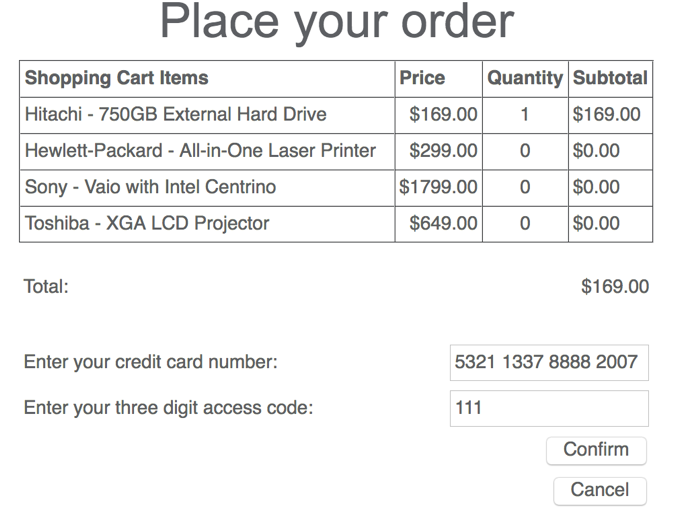
題目截圖：



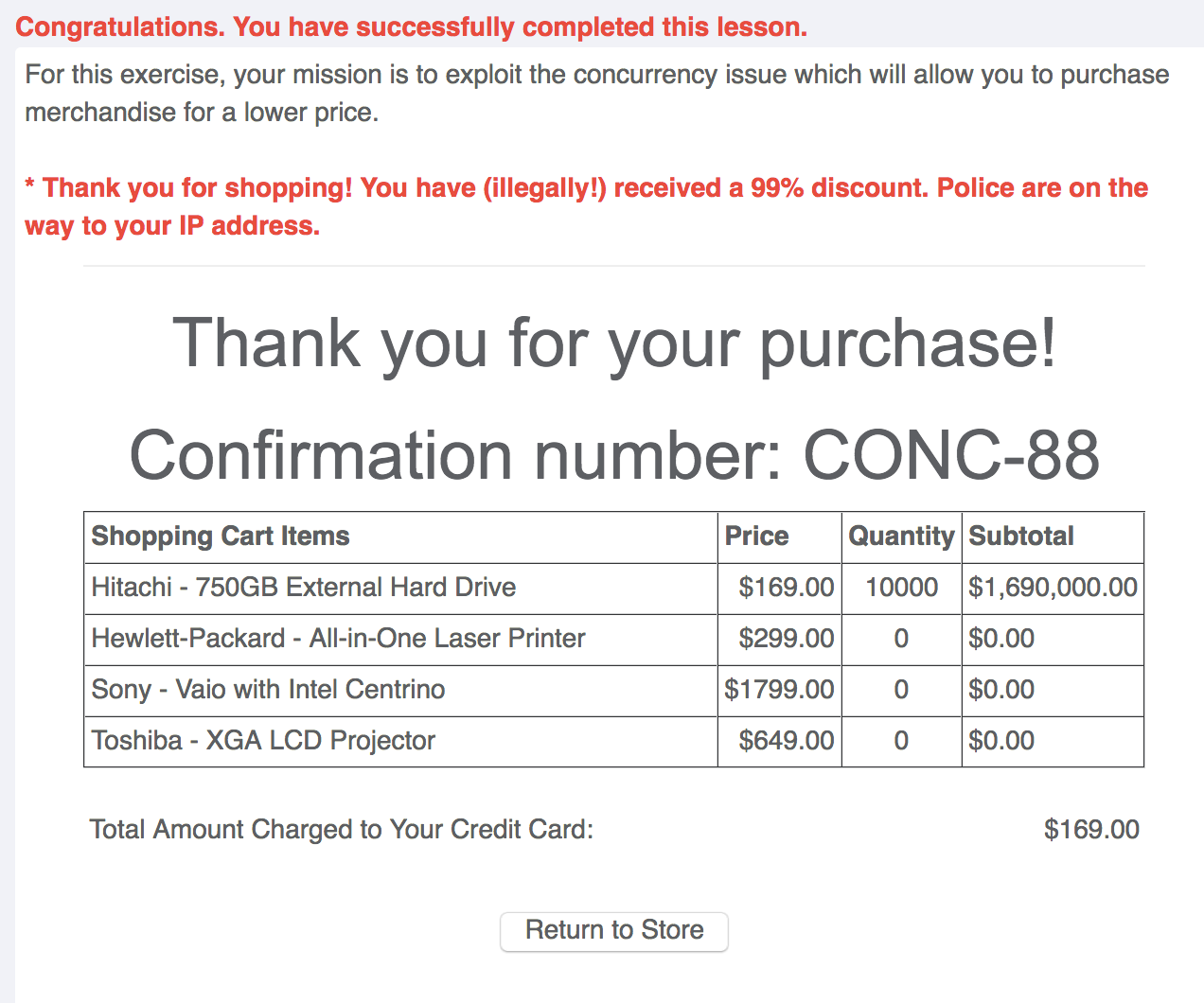
解題流程：

直覺思考是像是thread safety problem一樣，猜想subtotal是全域變數，透過兩個分頁同時對全域變數進行讀取，產生thread safety problem使得subtotal降低。但經過幾次嘗試後一直失敗，我想這應該不是解答。

於是我開啟兩個分頁，分頁A輸入Hitachi 1個並且按下purchase。

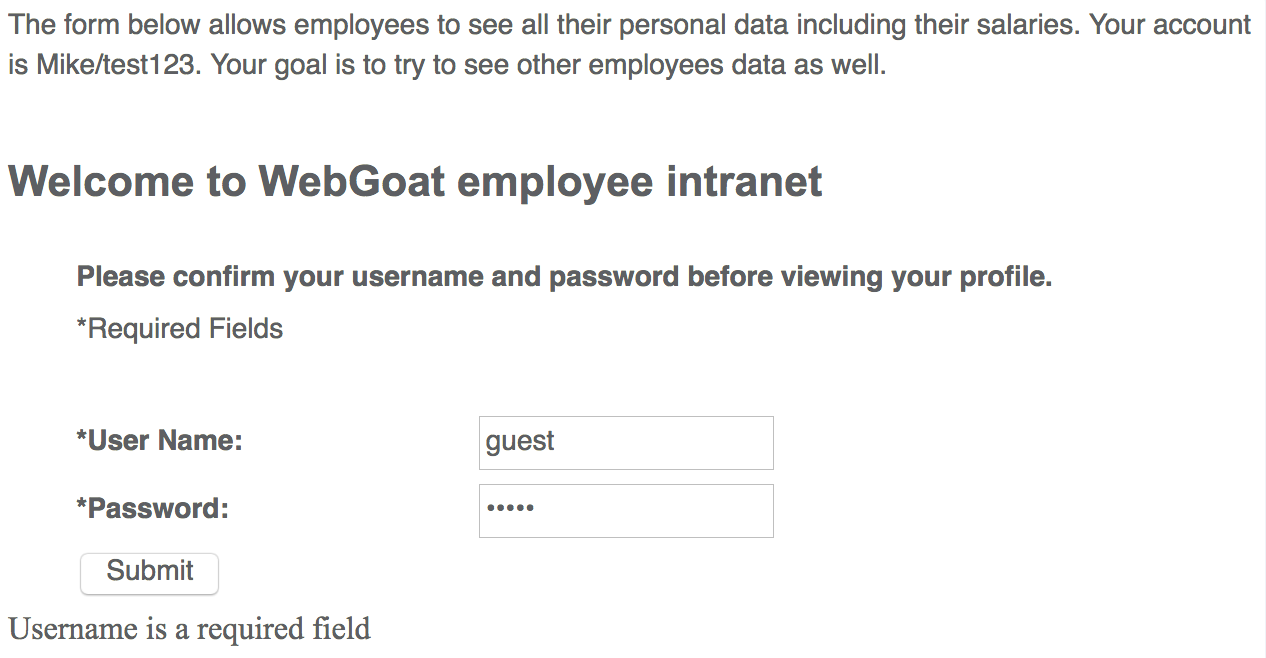


接著在分頁B輸入Hitachi 10000個並且按下purchase並且按下Update Cart（猜測quantity和subtotal為全域變數），然後在分頁A按下confirm。結果如下：



1. XPATH Injection

題目截圖：



解題流程：

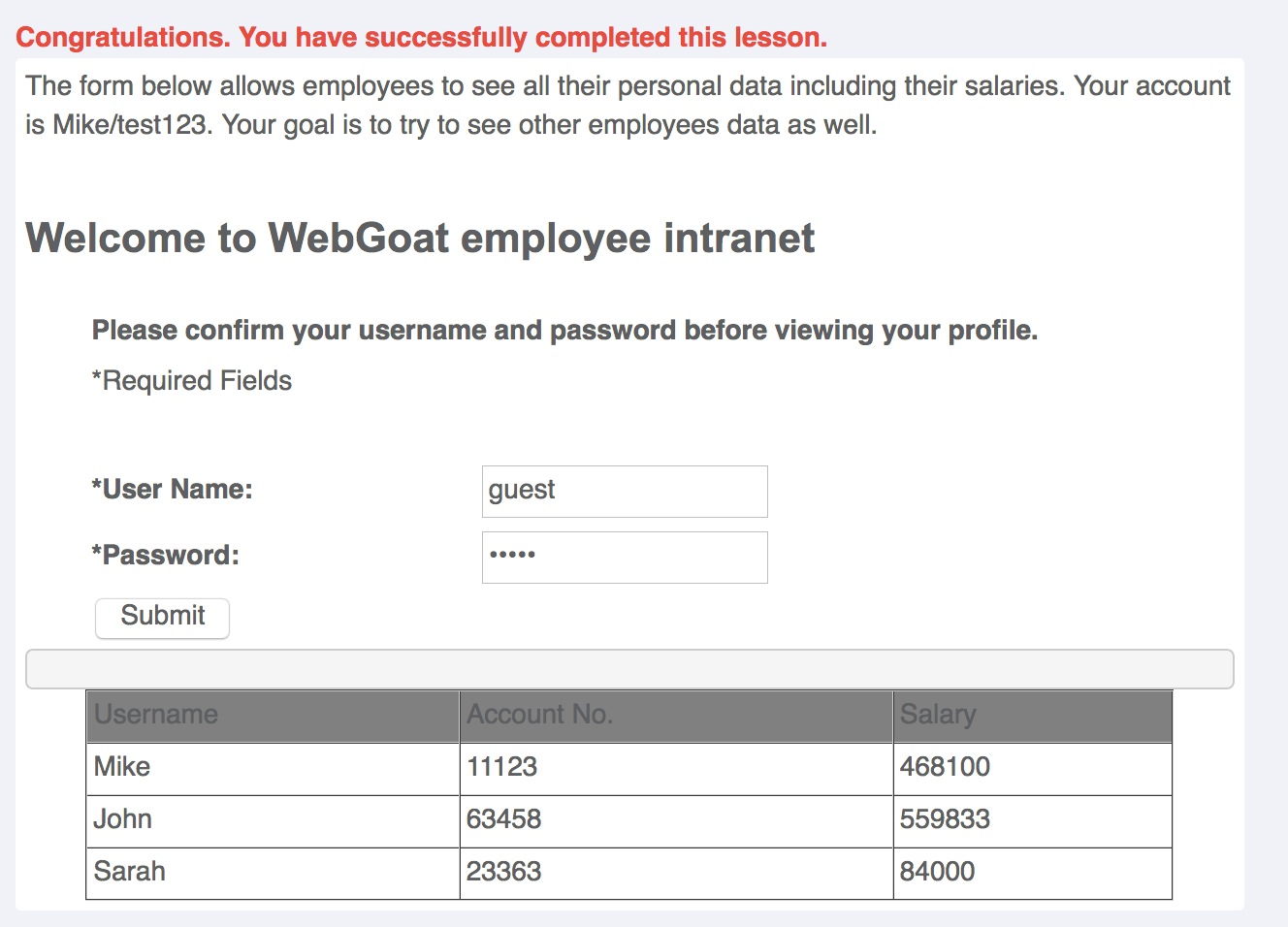
User Name : 1'='1

Password : test123’or ‘1’=‘1

猜測結果後端應該為類似：

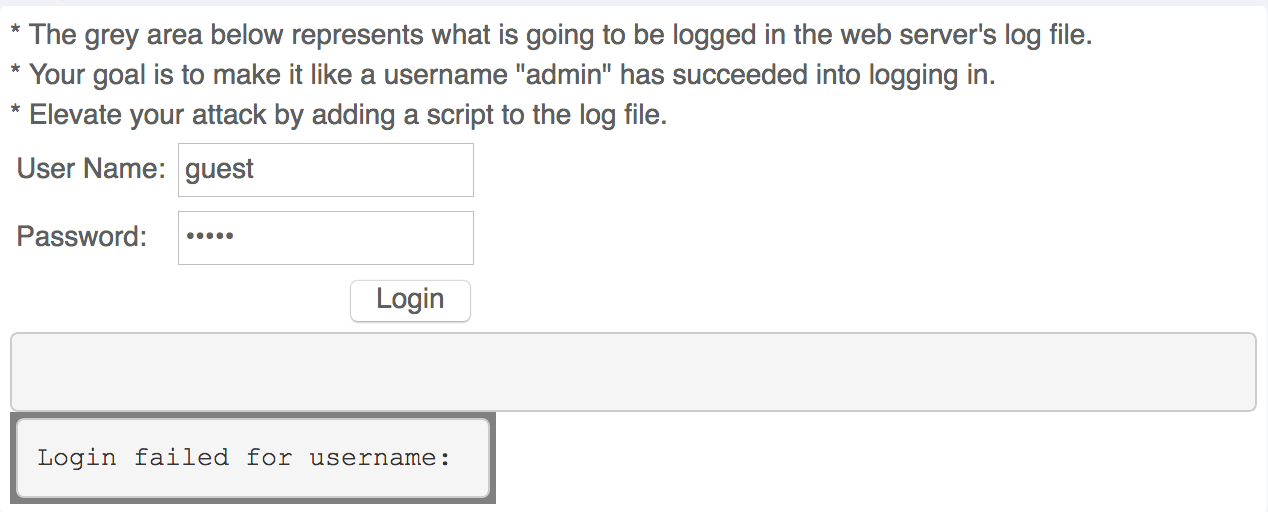
SELECT \* FROM USER\_TABLE WHERE UserName = ‘1'='1’AND Password = ‘test123’or ‘1’=‘1’

結果如下：



1. Log Spoofing

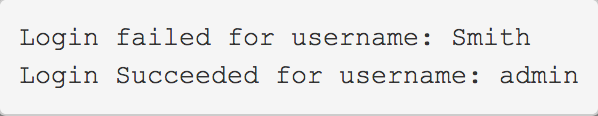
題目截圖：



解題流程：

題目寫說要 make it like a username “admin”has succeeded into logging in. 所以要使下方的灰色空格中印出：

Login succeeded for username: admin。由於我真的不知道要怎麼刪除Login failed for username:。於是看了一下答案，原來不用刪掉Login failed for username:，只要像下圖即可：

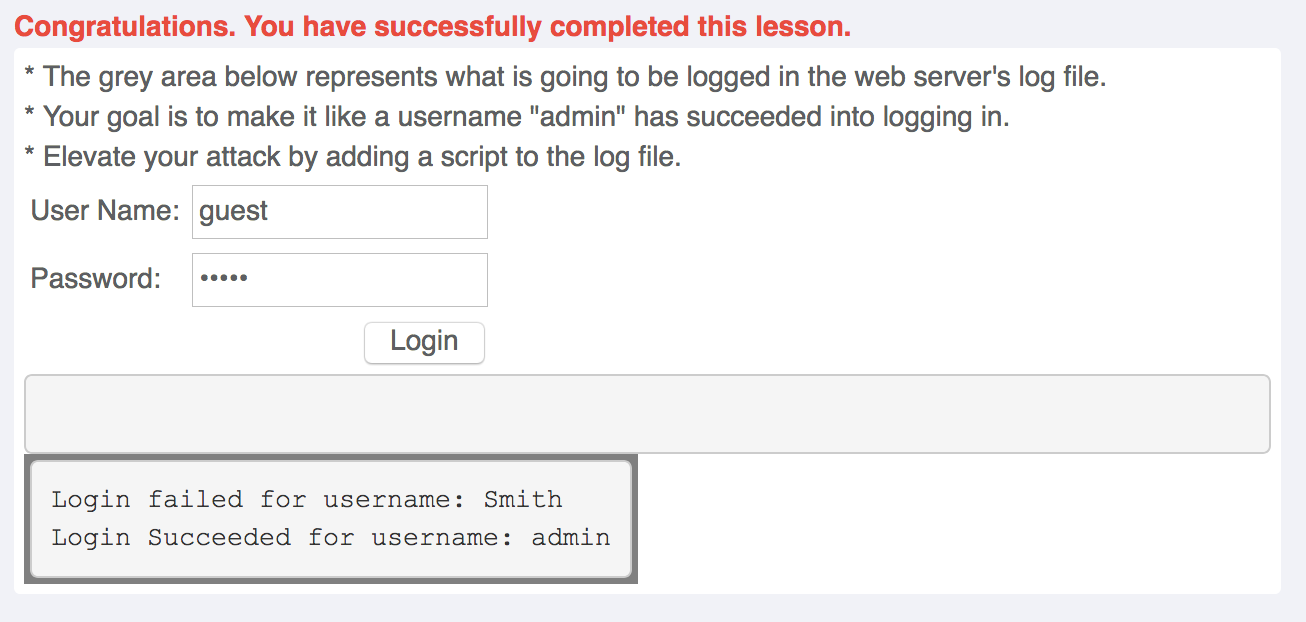


於是只剩下要思考如何換行：首先嘗試 \n 但失敗於是改用

%0a ，在User Name輸入：

Smith%0aLogin Succeeded for username: admin

結果如下：



由於numeric SQL injection 助教上課說過，故再加一題。