

اختبار

SQL Injection

1. الوصف: لديك صفحة تسجيل دخول تحتوي على الحقليين التاليين: اسم المستخدم وكلمة المرور. في الشيفرة الخلفية، يتم تنفيذ الاستعلام التالي للتحقق من صحة المستخدم:

```
SELECT * FROM users WHERE username = '$username' AND password = '$password';
```

المهمة:

- اشرح لماذا هذه الشيفرة عرضة لـ SQL Injection.
لأن القيم المدخلة من المستخدم يتم تمريرها مباشرة إلى الاستعلام بدون أي تحقق أو تنقية (Validation/Sanitization) هذا يعني أن المهاجم يستطيع إدخال شيفرة SQL ضارة مثل:

```
sql
' OR '1'='1
```

ليتجاوز التحقق ويسجل الدخول بدون بيانات صحيحة.

- اقترح طريقة لإصلاحها.
استخدام الاستعلامات المحضرة (Prepared Statements) التي تفصل البيانات عن الشيفرة، مثلًا في PHP باستخدام PDO:

```
php
$stmt = $pdo->prepare("SELECT * FROM users WHERE username = ? AND password = ?");
$stmt->execute([$username, $password]);
```

أو استخدام ORM يقوم بالتحقق تلقائيًا.

XSS

1. الوصف: لديك صفحة تعرض تعليقات المستخدمين دون تنقية المدخلات. يمكن لأي مستخدم إدخال تعليق يحتوي على شيفرة JavaScript:

```
<div><?php echo $comment; ?></div>
```

المهمة:

- اشرح لماذا هذه الشيفرة عرضة لـ XSS.
أن البيانات المدخلة من المستخدم (مثل `<script>alert('XSS')</script>`) تُعرض مباشرة على الصفحة بدون ترميز (escaping). هذا يتيح للمهاجم تشغيل شيفرة خبيثة على أجهزة المستخدمين الآخرين.
- اقترح طريقة لإصلاحها.

1. المدخلات (Input Sanitization): عدم السماح بشيفرات HTML/JavaScript في التعليقات.
2. المخرجات (Output Encoding): استخدام دوال لترميز النص مثل `htmlspecialchars()` في PHP :

```
php
echo htmlspecialchars($comment);
```

أو استخدام مكتبات تصفية مثل DOMPurify في JavaScript.

CSRF

1. الوصف: لديك نموذج لتغيير كلمة المرور يتم إرساله عبر طلب POST

```
<form method="POST" action="change_password.php">
  <input type="password" name="new_password">
  <input type="submit" value="Change Password">
</form>
```

المهمة:

- اشرح لماذا هذا النموذج عرضة لـ CSRF.

لأنه لا يوجد تحقق من أن الطلب فعلاً أتى من المستخدم الشرعي. يمكن لموقع خبيث إرسال طلب نيابة عن المستخدم بدون علمه إذا كان قد سجل دخوله مسبقاً.

- اقترح طريقة لإصلاحها.

إضافة رمز: CSRF (Token) يتم توليد رمز عشوائي في الجلسة وإرساله مع النموذج.
التحقق من الرمز في الخلفية: يجب أن يتطابق مع القيمة في الجلسة. مثال في HTML:

```
html
Edit Copy
<input type="hidden" name="csrf_token" value="<?php echo $_SESSION['csrf_token']
```

وفي الخلفية:

```
php
Edit Copy
if ($_POST['csrf_token'] !== $_SESSION['csrf_token']) {
    die("CSRF Detected");
}
```