# 🛡️ Secure Coding Assignment

\*\*Title:\*\* Identify and Fix Vulnerabilities in Given Code Samples

\*\*Module:\*\* Secure Coding

\*\*Due Date:\*\* 10 October 2025

\*\*Total Marks:\*\* 100

## 🎯 Assignment Objectives

This assignment focuses on helping you:  
- Identify security vulnerabilities in real-world code patterns.  
- Understand and explain exploit scenarios and impacts.  
- Apply secure coding practices to fix vulnerable code.  
- Justify your remediation choices based on OWASP guidelines.

## 📝 Task Description

You will be given multiple code snippets, each containing a common security flaw. Your task is to:  
1. Identify the type of vulnerability from one of the five specified categories.  
2. Explain how the vulnerability could be exploited (threat model).  
3. Fix the code using appropriate secure coding techniques.  
4. Justify your fix using secure coding principles and reference the OWASP Top 10.

## 🔐 Covered Vulnerability Categories (5 Only)

|  |  |
| --- | --- |
| Category | OWASP Reference |
| 1. SQL Injection (SQLi) | A03: Injection |
| 2. Cross-Site Scripting (XSS) | A07: XSS |
| 3. Insecure Deserialization | A08: Insecure Design or Object Injection |
| 4. Broken Access Control | A01: Broken Access Control |
| 5. Security Misconfiguration | A05: Security Misconfiguration |

## 📦 Submission Format

Submit a `.zip` archive containing:  
- `analysis.md` – for each snippet, include:  
 - Identified category  
 - Risk/exploitation description  
 - Screenshot (if tested)  
- `fixed\_code/` – a folder containing your secure versions of the given snippets  
- `justification.md` – your OWASP-aligned explanation of each fix

## 📊 Grading Rubric

|  |  |
| --- | --- |
| Criteria | Marks |
| Correct vulnerability identification (5 total) | 20 |
| Accurate explanation of exploitation | 20 |
| Secure fix implementation | 30 |
| Justification referencing secure principles | 20 |
| Code quality and clarity | 10 |
| Total | 100 |

## 🧠 Additional Notes

- Refer to https://owasp.org/Top10/ for OWASP categories.  
- Focus on practical fixes: use parameterized queries, input sanitization, access checks, etc.  
- Late submissions will incur penalties unless prior approval is given.