Secure Coding Review Project

Project Overview
This project is part of the **CodeAlpha Cyber Security Internship**. The task was to conduct a **Secure Coding Review** of a Python application, identify vulnerabilities, and provide secure remediation.

Methodology
1. **Language Chosen**: Python
2. **Tools Used**:
   - Bandit (static analysis)
   - Flake8 (linting & style check)
   - Manual code inspection
3. **Steps Followed**:
   - Reviewed source code for insecure practices.
   - Classified vulnerabilities by severity (High/Medium/Low).
   - Suggested fixes with best practices (OWASP Guidelines).

Findings
1. **Hardcoded Credentials** (High Risk)
2. **Insecure Input Validation** (Medium Risk)
3. **Weak Cryptography (MD5)** (Medium Risk)
4. **Missing Exception Handling** (Low Risk)

Recommendations
- Remove hardcoded credentials → use environment variables.
- Implement strong input validation to prevent SQLi & XSS.
- Replace MD5 with bcrypt/Argon2 for password hashing.
- Add proper exception handling.
- Follow OWASP Secure Coding Practices.

Code Example
Vulnerable Code (Before Fix)
```python
import hashlib
import sqlite3

DB_USER = "admin"
DB_PASS = "password123"

def login(username, password):
    conn = sqlite3.connect("users.db")
    cursor = conn.cursor()
    cursor.execute(f"SELECT * FROM users WHERE username='{username}' AND
password='{password}'")
    result = cursor.fetchone()
    if result:
        print("Login successful")
```

```
    else:
        print("Login failed")

def store_password(password):
    hashed = hashlib.md5(password.encode()).hexdigest()
    print("Stored (weak hash):", hashed)
```

Secure Code (After Fix)
```python
import bcrypt
import sqlite3

def login(username, password):
    conn = sqlite3.connect("users.db")
    cursor = conn.cursor()
    cursor.execute("SELECT password FROM users WHERE username=?", (username,))
    result = cursor.fetchone()
    if result and bcrypt.checkpw(password.encode(), result[0]):
        print("Login successful")
    else:
        print("Login failed")

def store_password(password):
    salt = bcrypt.gensalt()
    hashed = bcrypt.hashpw(password.encode(), salt)
    print("Stored (secure hash):", hashed)
```

Repository Structure
```
CodeAlpha_SecureCodingReview/
├── README.md
├── vulnerable_code.py
├── secure_code.py
├── Secure_Coding_Review_Project.pdf
└── presentation.pptx
```

Video Explanation
- Record a **2–3 min video** explaining:
  - Task objective
  - Tools used
  - Key vulnerabilities found
  - Before/After code examples
- Post it on **LinkedIn**, tag ‘@CodeAlpha‘, and include this GitHub repo link.

Author
Suraj Mishra

CodeAlpha Cyber Security Internship – Task 3: Secure Coding Review