Task 3: Secure Coding Review

Vulnerable Code

```
import sqlite3

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

Vulnerabilities

```
1. SQL Injection possible
2. Plaintext password storage
3. No input validation
4. Missing error handling
```

Fixed Secure Code

```
import sqlite3, hashlib

def hash_password(password):
    return hashlib.sha256(password.encode()).hexdigest()

def login(username, password):
    conn = sqlite3.connect("users.db")
    cursor = conn.cursor()
    query = "SELECT * FROM users WHERE username=? AND password=?"
    cursor.execute(query, (username, hash_password(password)))
    result = cursor.fetchone()
    if result:
        print("Login successful ■")
    else:
        print("Invalid credentials ■")
    conn.close()
```

Report & Best Practices

```
- Used parameterized queries to prevent SQL Injection
- Added password hashing (SHA-256)
- Safer input handling
- Improved reliability with secure coding practices
```