Task 3: Secure Coding Review (Python Example)

This task involves identifying security issues in code and improving it.

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()
    return result
```

Issues:

- SQL Injection vulnerability
- No password hashing

Secure Version:

```
import sqlite3
import bcrypt

def secure_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('utf-8'), result[0]):
        return True
    return False
```

Secure Practices:

- Use parameterized queries
- Hash passwords using bcrypt
- Validate all user inputs