Secure Coding Review Report

# 1. Programming Language and Application

• Programming Language: Python

• Application: Mock login and database interaction script

# 2. Tools Used

• Manual Code Review

• Bandit - Python static analyzer

# 3. Findings

|  |  |  |  |
| --- | --- | --- | --- |
| Issue | Location | Severity | Description |
| Use of eval() | Line 10 | HIGH | Allows arbitrary code execution |
| SQL Injection Risk | Line 8 | HIGH | Query uses unsanitized input |
| Hardcoded Credentials | Line 4-5 | MEDIUM | Secrets should not be hardcoded |
| No Input Validation | Line 12 | LOW | Inputs should be validated (e.g., isdigit) |

# 4. Recommendations and Best Practices

• Avoid using dangerous functions like eval() or exec().

• Use parameterized queries for database operations to prevent SQL injection.

• Store credentials in environment variables or configuration files, not directly in code.

• Validate all user inputs before processing.

# 5. Remediation Steps

• Replace eval() with safe alternatives like formatted strings.

• Modify SQL queries to use parameterized formats (e.g., cursor.execute(query, params)).

• Replace hardcoded credentials with secure storage using os.getenv().

• Validate numerical inputs using str.isdigit() or type casting with error handling.