UNIVERSITY OF TWENTE.

Module 5 - Computer Systems (2021-22) Project

Testin

g-SecuritybyDesignChecklist

Team ID: 17	**Team Members:** Hein Huijskes, Julia van der Geest, Ujjwal Dodeja, Jose Gavilanes, Maouheb Bessi, Mengmeng Li
Project Name: ChessMate	**Mentor(s):** Ramish Bhutto, Wenjie Zhao

Instructions:

1. Refer to the below table. All the mentioned points are mandatory to perform for your application except point no. 4.
2. You should consider at least 2 vulnerabilities for each criteria given in Column 'B', except point no. 4, 6, and 7.
3. The mitigation plan/solution should be considered for every identified vulnerability.
4. Make sure to review the document with your team members and mentor(s) before final submission.
5. This checklist should be in lined and submitted along with the Software Testing document.

Points	Source Code Review, Static and Dynamic Application Testing	Identified Vulnerabilities for testing (Name them)	Put tick ✔(if you have completed a the points as mentioned in Column 1.
1	Application security vulnerabilities (e.g. Access Control, Injection, Authentication, Cross Site scripting, etc.)	SQL Injections were potentially an issue for our project, by using prepare statements in our SQL queries, we eliminated this threat. Access control is not an issue, since we require proper credentials to login and to play the game, the user does not have any rights on modifying information so far. Authentication happens based on the user's credentials, this gets checked against the hashed password with salt. If this is correct, the user can continue to the main screen.	✔
2	Weak security in functions (e.g. old encryption techniques, Hashing, Privileges assigned, Function error, etc.)	We use SHA-512 hashing for the password and salt. The salt gets generated using the UUID class. There is only a vulnerability if an attacker finds a way to crack the SHA-512 hashing algorithm. Only hashed versions on the password get compared in the code.	✔
3	Duplicate/unnecessary functions	We used several predefined libraries and derived functions only for specific tasks. Each function added by us plays a specific and significant role in the functioning of the entire system. There are no unnecessary functions or pieces of code in our system.	✔
4	Analyzing Program (e.g. computation time, power	-	

	consumption, etc.) **(Optional)**		
5	Address the remaining vulnerabilities of your application (manual)	voice detection is not sensitive and fast;	✕
6	Make a mitigation plan/solution by listing down the vulnerabilities	There are some vulnerabilities such as: -The voice command might not be the one the player gave because the speech recognition library works better with words or sentences rather than commands like A1 A2, so a solution is to use words instead of letters like Echo 1 instead of E1. -It can happen that a move was misunderstood because of bad pronunciation or the recognizer could not get the right command, so to address this issue an undo move function/command is a good solution for that. -There is also the possibility a player can cheat their opponent by making moves on behave of their opponent, which also an undo functionality can address this issue, another solution could be implement voice recognition so that a move will be performed just by the player whose turn it is	✕
7	Review with your team members and approve by your mentor(s).	-	

Team members reviewed:

(Member 1, Yes), (Ujjwal Dodeja, Yes), (Maouheb Bessi, Yes),(Jose Gavilanes, yes)(MengmengLi, Yes), (Julia van der Geest, Yes), (Hein, Yes)

Mentor(s) reviewed and verified:

(Mentor 1, Yes), (Mentor 2, Yes), …

Prepared by:

Dipti K. Sarmah (Project Coordinator)