Module 5- Computer Systems (2021-22)

Project

UNIVERSITY OF TWENTE.

Testing-Security by Design Checklist

Team ID: Team 30	**Team Members:** Jan van Zwol, Vo Nhat Minh, Tran Duc Duc, Marjolein Bolten, Ho Hoang Phuoc, Daan Velthuis
Project Name: Ball on the wall	**Mentor(s):** Puru Vaish & Venelina Pocheva

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 inline 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 all the points as mentioned in Column 1.	Remarks, if any
1	Application security vulnerabilities	- Duplicate usernames - Weak password accepted - Infinite trials to brute force the password - SQL injections by malicious users - The hashed passwords are already in rainbow tables and thus easy to crack		
2	Weak security in functions	- Insecure hashing algorithm - Camera is not protected while it captures the game screen.		
3	Duplicate/unnecessary functions	- Deprecated functions are still in used		
4	Analyzing Program (e.g. computation time, power consumption, etc.) **(Optional)**	- No vulnerabilities		

5	Address the remaining vulnerabilities of your application (manual)	- non-admin users can access the database to modify it - non-admin users can try to login as admin - admin sells users data for money - users data stays too long in the database - Data can be abused due to a lack of clear legal documents - people can look at the database outside of the game environment if saving on pi of locally - an attacker can access the camera and view the personal environment of the user		
6	Make a mitigation plan/solution by listing down the vulnerabilities	-		
7	Review with your team members and approve by your mentor(s).	-		

Team members reviewed:	Jan van Zwol	Yes
	Vo Nhat Minh	Yes
	Tran Duc Duc	Yes
	Marjolein Bolten	Yes
	Ho Hoang Phuoc	Yes
	Daan Velthuis	Yes
Mentor(s) reviewed and verified:	Puru Vaish	-
	Venelia Pocheva	-

Mitigation	**Plan to solve this**
Non-admin users can access the database to modify it	Only giving 1 account admin rights and the username and password of this account will be unique and the password will be changed every month.
Non-admin users can try to login as admin	Only giving 1 account admin rights and the username and password of this account will be unique and the password will be changed every month.
Admin sells users data for money	There will be legal documents to disapprove admin abuse.
Users data stays too long in the database	We will delete the user data on request of the user.
Data can be abused due to a lack of clear legal documents	There will be legal documents to prevent abuse of data.

People can look at the database outside of the game environment if saving on pi of locally	We will make a separate database phpPgAdmin which is not connected to the Raspberry Pi.
An attacker can access the camera and view the personal environment of the user	The camera will only be turned on when the user is playing the game. When the camera is on, only the coordinates from the ball will be used and the images the camera creates will not be sent along. This way it becomes impossible to access it through the internet. We will also never store the input from the camera.
Insecure hashing algorithm	We will use the Bcrypt hashing algorithm which is one of the most secure algorithms. The passwords will be hashed in SHA-512 with the unknown random iteration and salt and Bcrypt is intended to slow down the hashing check.
Infinite trials to brute force the password	The login process will be artificially made to take longer (with Bcrypt built-in function) so it will take a very long time to brute force a lot of different passwords.
Duplicate usernames	During registration, we check if the username already exists.
Weak password accepted	During registration, we check for unsafe passwords by checking for lower, upper case letters and numbers and length of at leat 8 characters
The hashed passwords are already in rainbow tables and thus easy to crack	We will add random salts to the hashing of passwords using Bcrypt random hashing function.
Password longer than 1023 characters	The maximum length of the password input field will be 30 characters.
SQL injections by malicious users	We will use parameterized SQL queries

Module 5- Computer Systems (2021-22)

Project

UNIVERSITY OF TWENTE.

Software Testing Document (STD) Template

Team ID: 30	**Project Name:** Ball on the wall
Team members: Jan van Zwol, Vo Nhat Minh, Tran Duc Duc, Marjolein Bolten, Ho Hoang Phuoc, Daan Velthuis	**Mentor (s):** Puru Vaish & Venelina Pocheva

Instructions:

1. Refer to the below table and complete all the sections with clarity.
2. Select those test strategies that are applicable to test your application.
3. Make sure to refer to the "Development-Security by Design Checklist" to see the possible vulnerabilities in your application.
4. Feel free to add features and test cases in the table that are essential to test your application.
5. You can use Selenium, SonarQube, and/or GitLab CI/CD to perform source code review, static and dynamic application testing, etc.

Test Strategy	Date (When did you perform the testing?)	Process/Function (Features to be tested)	Test Case	Step	Description	Status (Passed/Failed /Open)	Expected Results	Actual Result	Mitigation plan/Solutions	Review on the Mitigation plan (Passed/Failed)	Remarks on the Failed mitigation plan
Application test	01/11/2021	Login	Logging in with correct password and username	1	The correct userID and password should be entered.	**Passed**	User should access the home page.	The user will go to the next page (home screen).	Does not apply	Does not apply	…
Application test…	01/11/2021		Logging in with incorrect password or username	2	The incorrect username and password would be entered.	**Passed**	You get an error message that you have an invalid combination of username and password.	An error is displayed that the password is incorrect.	Does not apply	Does not apply	…

Application test	01/11/2021	Registering	Register with new username and accepted password	1	A new username with an accepted password is entered.	**Passed**	The new user is registered in the database and the user will be able to go back to login with new account	The user got a message saying: 'registration successful, now go to the login page'. Also the new account is stored in the database.	Does not apply	Does not apply	
Application test	01/11/2021		Password Acceptance Check	2	The length of the password should be at least 8 characters with 1 symbol, 1 uppercase, 1 lowercase and 1 digit.	**Passed**	The user gets an error message when trying to create an account with a password which does not hold for all the conditions and should specify which condition is not met.	An error message is shown including which condition is not met for the password.	Does not apply	Does not apply	
Application test			Duplicate Username Check	3	It should not be possible to register a user with an already existing username.	**Open**	The user will get an error message when trying to create an account with an already existing username.				
Application test	01/11/2021	Security	SQL injection protection	1	Attackers try to input SQL malicious code through the login/register option to get all content of the database(username, score, password) or to corrupt it.	**Passed**	The user input with SQL injection will not be executed and an error message to show that there is invalid character	An error message is shown that there are invalid characters and the SQL injection is not executed.			
Application test			Slowing down password check	2	It should take more time for attackers to do a brute force to crack the passwords.	**Open**	It takes 3-5s to login.				
Application test + Manual Test	01/11/2021		Hashing + Salt password	3	The password should not be stored as plain text in the database	**Passed**	The password will be hashed using Bcrypt with salt before being stored in database	The password is hashed with salt using Bcrypt before being stored.			
Manual test	2/11/2021		Non-admin user higher privilege access check	4	Non-admin users should not be able to get access to the admin options.	**Failed**	If a non-admin user is logged in he should see another interface (without the option to modify the database of the admin user). It is impossible to make a request yourself.	A non-admin user does see another interface, without the option to modify the database of the admin user. However, it is possible to make a request yourself by altering the	We want to only store the binary file, such that you can not access the source code anymore.		

								source code, which contains the login credentials of the database.			
User acceptance test			Privacy of users	5	Check whether the images recorded by the camera’s aren’t saved in the database	**Open**	If the admin looks at the database there should be no saved images.				
Manual test		Admin user functions	Admin user options	1	Admin users should be able to get access to the database with the option to delete and reset the database.	**Open**	Admin users see the option to delete and reset the database.				
Application Test			Delete Test	2	If a user's data is deleted all his data should be deleted.	**Open**	The user's account and scores are removed from the database.				
Application Test + Manual Test			Reset Test	3	If the admin clicks on reset then the scoreboard will be reseted.	**Open**	There are no more scores in the database.				
Manual test	01/11/2021	Hardware functions	Side camera test	1	The side camera should be able to see when the ball bounces on the wall.	**Passed**	The side camera gives a signal when the ball hits the wall.	The program sends a signal when the ball hits the wall.			
Manual test			Front camera test	2	The front camera should be able to see where the ball hit the wall.	**Open**	The front camera gives the coordinates of the ball where it hit the wall.				
Manual test			Delay test	3	The delay of input from the camera to update the game should be less than 3 seconds.	**Open**	After the ball hits the wall, the game will be updated within 3 seconds if needed.				
Application test + Manual Test		Game functions	Score test with cowboy	1	The user should get bonus points for hitting a cowboy.	**Open**	When the user hits a cowboy, it gains points.				

			Score test with cow	2	The user should get minus points for hitting the cow	**Open**	When the user hits a cow, it loses points.				
Manual Test			Interface test with cowboy	3	When a cowboy is hit it should disappear and a new cowboy is generated	**Open**	When the user hits a cowboy, it should be removed and a cowboy should be generated				
			Interface test with cow	4	When a cow is hit it should disappear and a new cow is generated	**Open**	When the user hits a cow, it should be removed and a cow should be generated				
Application test			Trial test	5	Everytime you miss a cowboy you will lose 1 trial.	**Open**	When the user misses a cowboy, the trial count will be decremented with one.				
Manual Test			Endgame test	6	The game will end when the number of trials becomes zero.	**Open**	When the number of trials becomes zero, we will see the game over message with the score.				
Application test + Manual Test			Scoreboard test	7	Users will get the list of top 10 highest scores.	**Open**	Users will get the list of top 10 highest score.				
Application test + Manual Test			Highscore update test	8	If a user gets a high score that is high enough to be in the scoreboard then the score board should be updated correctly.	**Open**	User will see his score and rank in the scoreboard.				
	2/11/2021		Object generation test	9	The object should be generated within the game screen and the number of cows and cowboys is reasonable.	**Passed**	We can see that the new object should be generated within the game screen and the number of cowboys will be between 1 and 3 and the number of cows will be between 1 and 3.	The objects are generated within the game screen and there are always between 1 and 3 cows and cowboys.			

Static application testing		Source code review	Source code syntax test	1	The source code should follow conventions and best practices.	**Open**	The source code follows conventions and best practices.				

Note: Refer to the following documentation on GitLab and SonarQube for clarity-

1. Source Code review with SonarQube: **https://docs.sonarqube.org/latest/**
2. GitLab integration with SonarQube: **https://docs.sonarqube.org/latest/analysis/gitlab-integration/**
3. SonarQube (Static Application Testing): **https://www.sonarqube.org/features/security/**
4. Gitlab (Static Application Testing): **https://docs.gitlab.com/ee/user/application_security/sast/**
5. GitLab (Dynamic Application Testing): **https://docs.gitlab.com/ee/user/application_security/dast/**

Prepared by:

Dipti K. Sarmah (Project Coordinator)