Module 5- Computer Systems (2022-23)

Project

UNIVERSITY OF TWENTE.

Software Testing Document (STD) Template

Team ID: 4	**Project Name:** PiSec
Team members: Thijs Frauenfelder, Katy Radzkova, Ayolt ten Have, Victor Zugravu, Frank Bosman, Mikus Vancans	**Mentor (s):** Priya Naguine & Radu Basarabá

Instructions:

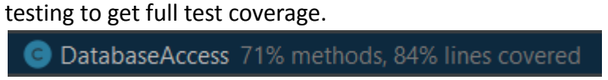
1. Refer to the below table and complete all the sections with clarity.
2. Select those test strategies that are applicable to test your application.
3. Make sure to refer to the "Development-Security by Design Checklist" to see the possible vulnerabilities in your application.
4. Feel free to add features 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
Unit testing	1-11-2022	Database	Correct storage of data in the database	1	The data that is extracted to the database is the same data that have been written in there	**Passed**	account data is stored correctly according to the unit test	account data is stored correctly according to the unit test			
Manual testing	21-10-2022	Motion detection	Processes awaken when motion detection is triggered (capture video-> store the file path of the video in the database->send alert notification to the user)	1	When the motion detector is triggered the corresponding signal appears in the console.	**Passed**	When the motion is detected, the line “Motion” appears in the console	When the motion is detected, the line “Motion” appears in the console			
				2	The alert information(date and time, path to the captured video) is stored in database after detection	**Passed**	Alert information is stored in the database after motion detection	Alert information is stored in the database after motion detection			
				3	Email and WhatsApp notification is sent to the user after detection	**Passed**	an email and whatsapp notification is sent after motion detector was triggered	notifications are sent after motion is detected.			

Manual testing	31-10-2022	Authentication	User login into personal web interface	1	The user is only able to log in with the correct login details	**Passed**	After correct login details (email and password) are filled into the fields on the web interface, the user is redirected to his personal home page.	The user is logged in and redirected to their personal home page.			
				2	The user is not able to use the service if invalid login details are entered	**Passed**	After incorrect details were entered, the web application gives a corresponding alert.	Trying to log in with invalid login details showed “The details are incorrect. Try again” message to user and remained on login/signup page			
Manual testing	7-11-2022	Structural Integrity of system and its vital components	The system is transported in a bag over a regular period of time	1	As a portable security system it is important that the physical product is able to remain intact throughout use and transport	**Passed**	The casing of the system would remain intact and protect the interior components	The casing remained intact and no damage was observed			
Manual testing	5-11-202	Functionality of System restricted due to physical obstructions	Interference of the camera view	1	The camera should continue to work properly in case of physical obstacles (aka, being covered or placed in the dark environment)	**Failed**	The camera captures a clear video of the intruder	The camera captures the video,however there is no clear view of the object that triggered the motion detector	Adjust the setting of the camera to get the video corresponding to the actual environment	**Failed**	This feature is not included in the functionality of the project, so the user should be responsible to provide a clear view to the camera with no obstacles and enough light.
Application testing	31-10-2022	Security	Protection from SQL Injection related attacks	2	Preventing malicious SQL injections into the web interface text fields	**Failed**	Potential SQL injections are prevented by using prepared statements	SQL injections are **not** prevented	i) Check whether input sanitisation code is implemented correctly. ii) Check if prepared statements for queries are used.	**Passed**	
End to end testing	7-11-2022	Navigation	web page, navigation.	1	An automatic user is logged in and automatically navigates through the site checking if it is able to do so	**Passed**	The automatic user is able to navigate along the page	The automatic user was able to navigate everywhere			
End to end testing	7-11-2022	Authentication	Automatically User login into personal web interface	1	Enters the wrong data in multiple ways and checks the error messages	**Passed**	For wrong or invalid data an error notification is expected.	The same			

				2	When the right data is entered, th automatic user checks whether the page was redirected	**Passed**	For correct data entered, a redirect to the home page and correct cookies are expected.	The user is redirected to the dashboard page			
End to end testing	7-11-2022	Authorization	Authorization of the different webpages	1	Use an automatic user to visit every page of the website without logging in and testing if it is being redirected to the login page	**Passed**	The automatic user is redirected from every page to the login page.	The user is redirected back for every page.			
End to end testing	7-11-2022	User interactions	web page, Interactions	1	The automatic user tests all the input fields, buttons and sees if the action is as it was meant to be.	**Passed**	All the input fields, buttons worked as expected in the tested browsers, Chrome, Edge, Electron, Firefox	All the all the input fields, buttons worked as expected in the tested browsers, Chrome, Edge, Electron, Firefox			
End to end testing	7-11-2022	Loading of user data from the backend	web page, loading from backend	1	The automatic user navigates to every page and looks if the data is loaded correctly	**Passed**	The data is retrieved and displayed to the user.	The data is retrieved and displayed to the user.			
Application testing	31-10-2022	Security	Protection against cross site scripting	1	Prevent cross site scripting attacks	**Passed**	Scripts input by users get put into the webpage as text, not as html code, so it can never work as a script	Scripts are input as a text, so cross site scripting is not possible			
	31-10-2022		Hashing passwords frontend	2	Passwords should be hashed in the frontend so that a user's password is never transferred over the network	**Passed**	Passwords are hashed when they arrive in the backend from the frontend	Passwords are hashed when they arrive in the backend from the frontend			
	31-10-2022		Salting, peppering and hashing passwords in the backend	3	Passwords should be salted, peppered and hashed in the backend so that an attacker can not crack passwords with the use of rainbow tables	**Passed**	Password are peppered, salted and hashed before being stored in the database along with their salt	Password are peppered, salted and hashed before being stored in the database along with their salt			

Code Coverage: We use unit tests to test our functions which write or access the database. These functions are in DatabaseAccess and as seen we have a coverage of 71% of the methods with our tests. This includes all the functions which access data or write data to the database but because we can't check the functions which make the connection to the database the coverage is not 100%. Our other parts of the code we check with manual testing and end to end testing to get full test coverage.



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)