CS1IAD Portfolio 3 Report

Name: Amariah Rigodon. Student ID Number: 210069828

Basic information

Your Website URL: *http://210069828.cs2410-web01pvm.aston.ac.uk/login.php*

Source code link (if applicable):

One registered user account:

Username	**Password**
Amariah	abc

Brief Description of Technologies and Structure

I have designed my website using pure PHP. I used some built in PHP functions to make the system more secure (for example, to prevent html injections). I also utilised client-side technologies such as html, css and javascript. HTML allowed me to display information to the user, css enabled me to design the webpage and make it appealing, and lastly, I used Javascript to output some messages, and implement security features.

Implementation of the required functions:

Functions	**Main Source File(s)**	Working	**Notes**
P1. View the basic information of all cvs	*cv.php*	*yes*	
P2. Click one CV in the list to view the CV details	*cv.php, AllCVdetails.php*	*yes*	*There's a hyperlink in cv.php. When it's clicked, some code in AllCVdetails.php is executed.*
P3. Search all CVs using first name, last name or key programming language	*cv.php, AllCVdetails.php*	*yes*	*The search bar and radio boxes are in cv.php. Based on which is selected an on what user searches, some code in AllCVdetails.php is executed.*
P4. Register to become a registered user	*registrationForm.php*	*yes*	
R1. Log in the system	*login.php*	*yes*	
R2. Update your CV	*updateCV.php*	*yes*	
R3. Logout the system	*logout.php*	*yes*	

Implemented security features

Features	**Main Source File(s)**	**Notes**

Authentication	*login.php, updateCV.php*	*Users must input their username and password when logging into the system, and to update their CV. If they're not registered or their passwords don't match that in the database, they will receive an error message.*
Authorisation	*updateCV.php*	*Only user can update their cv as they must input their password first. password_verify() is used to verify passwords.*
Form validation	*login.php, updateCV.php, registrationForm.php*	*The minimum and maximum length is set for each input box. Some fields are required. Emails must contain an '@'. I validated using html.*
Handle injections (SQL/HTML)	*updateCV.php, login.php, registrationForm.php*	*I used the filter_var() function to remove all characters that should not be in an email address, for example <>, (),//. This also helps prevent html injection. The Htmlentities() function translates characters to HTML entities and prevents harmful HTML codes from being inserted into sites.* *I prevented SQL injections by using prepare() SQL statements with bound parameters, and the execute() function. Prepared statements make sure that the intent of a query cannot be changed, despite SQL commands being inserted.*
Hash password	*registrationForm.php, updateCV.php*	*Password is encrypted before being sent to the database.*
Cross-Site Request Forgery	*Portfolio3JS.js*	*I used a website for guidance. The link: https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html*

Notes to examiners (if applicable)

I changed the username and password from root, '' to u-210069828,w6QXB1XXCmatOQG when deploying the website.