|  |  |  |  |
| --- | --- | --- | --- |
| Module Code | DWD 507 | Module Title | Security and Testing |
| Student ID | 3201807 | Name | Jack Bradley-Coombes |

**Section 1 Web application security [35 marks]**

Considering your recommendations from *DWD 507 Assessment 1* *Antique Mall* had appointed you to further modify the website code and test it. They have given you full access to the website code including the database and admin panel. Your task is to demonstrate how you can preventfurther attacks on the *Antique Mall* website.

**Tasks**

1. Re-write the website code for the following:
2. To prevent SQL injection attacks. [6 marks]
3. Enforce users to create a strong password. The password must be sent using a secure method and encrypted before sending it to the server. [6 marks]
4. Secure admin panel. [3 marks]
5. Add your code to the website and test your updated website. Provide the evidence for the following:
6. A SQL injection attack no longer works [6 marks]
7. Users are required to create a strong password that should be encrypted and sent to the server using a secure method. [6 marks]
8. Only authorised users can access the admin panel. [3 marks]

Provide a summary of completed tasks in the format given below (add more rows/columns if required):

|  |  |
| --- | --- |
| **SQL injection attack** | |
| **Filename** | listing.php |
| Old Code  (if applicable) |  |
| New Code |  |
| Testing result summary (what is being done & why and the outcome) | To prevent back-end SQLI attacks, I added code to prepare, bind and execute the statement to protect from SQLI attacks, after entering this code, no errors were sent back. |
| Evidence of testing and result updated website | Page working without errors:  Working properly |
| **Filename** | contact.php |
| Old Code  (if applicable) |  |
| New Code |  |
| Testing result summary (what is being done & why and the outcome) | Before editing code, when entering ' OR '1'='1 I get a fatal error. After editing code by adding the use of prepared statements and biding parameters, it sends the line as a message instead of viewing it as code, meaning the page is safe from SQLI attacks. |
| Evidence of testing and result updated website | Proof SQL attack no lonnger works:  Before:    After: |

|  |  |
| --- | --- |
| **Use of strong password** | |
| **Filename** | register.php (requirement of a strong password and password hashing) |
| Old Code  (if applicable) | Hashed password pre-edited:    Strong password pre-edit: |
| New Code | **Password Hashing:**    **Strong Password:** |
|  |  |
| Testing result summary (what is being done & why and the outcome) | After registering an account, the saved password in the php database is an encrypted string of characters. Passwords are restricted to not allow passwords shorter then 6 characters, not including special characters, or that are too common, like “password” or “admin”. |
| Evidence of testing updated website | Encrypted password:    Proof of login system restrictions: |
| **Filename** | login.php (verifying hashed password) |
| Old Code  (if applicable) |  |
| New Code |  |
| Testing result summary (what is being done & why and the outcome) | Before changing the code, the login page wouldn’t accept my password, afterwards it accepted the password, and I could log in to previously registered account. |
| Evidence of testing updated website | Before:    After:    (account successfully logged in) |

|  |  |
| --- | --- |
| **Secure admin panel** | |
| **Filenames** | List all the files lacking admin verification  Index, orders, manage users, delete-image, listing-gallery, single order |
| New Code | require\_once(\_\_DIR\_\_ . '/includes/seccheck.php'); |
| Testing result summary (what is being done & why and the outcome) | I looked through all admin files for the code that links the code to the admin php script. After entering missing code on said pages, a restricted user can no longer access the page. |
| Evidence of testing updated website | Result of user without staff role trying to access admin pages: |

1. Discuss the security measures for secure online transactions in the table given below. [5 marks] (150-200 words)

|  |
| --- |
| **Secure online transaction** |
| Discuss requirements to prevent violation of PCI DSS for Antique Mall. As part of your discussion clarify if Antique Mall is PCI DSS compliant already or if they need to make changes to become PCI DSS compliant. |
| To make the Antique Mall website PCI DSS complaint, you should get an SSL/TLS certificate when hosting the website. This will ensure data entered on the website is encrypted and safe, keeping any sensitive data entered secure. Doing this will also make the website look more legitimate to customers, increasing the appeal of the website. The website doesn’t currently have this due to being locally hosted, but once the website is put onto a full server host this is a must. Along with encrypting information, its also good to install a firewall system to keep malicious traffic out. An antivirus will act as an extra layer of security to protect from malware and threats targeting the website. Ensuring all software are maintained and updated is a must, to prevent attacks from new threats. Performing regular monitoring to check for issues, and penetration tests to test for vulnerabilities will keep the website secure. |

**Section 2. Web application testing [35 marks]**

1. Write a test plan to test the following functionality of the *Antique Mall* website and provide the testing evidence.
   1. **All external/internal links-test Navbar, Social media icons, all buttons and anchor tags**

URLs tampering works to get from one page to another but restricted pages like admin panel are unable to be URL tampered into. All buttons and links appear to work as intended. The social media buttons don’t link to social media pages, and instead link to vision website, however I assume this is because the website does not have social media accounts.

* 1. **Login system – test registration and login page with empty/valid/invalid inputs, check the user roles after successful login**

Account registering system doesn’t allow users to enter empty fields, or unsecure passwords, by restricting them to be at least 6 characters long, contain a special character like #, $ or, !, and exclude common passwords like “password”. The registration form doesn’t allow SQLI inputs. Login form also doesn’t allow invalid input, nor SQLI attacks. The login form successfully pulls account information from the database and allows the user to login. Apon creating an account, the password is encrypted on the database, and the user is set to “not staff”.

* 1. **Pages- test home page, single listing page and the contact us form with empty/valid/invalid inputs**

Listing pages for lamp and guitar have a large empty area on them, possible missing image. Contact us form doesn’t allow SQLI or empty inputs. Valid inputs are sent into the database.

* 1. **Order processing- test add to Wishlist, a duplicate entry in the Wishlist table, Add to Cart, validate billing inputs ?**

Wishlist and Cart works as expected. You cannot add the same item to wish list/ cart twice, as the wish list/cart option is replaced with remove from wish list/ remove from cart button once on the wish list/cart. Billing section inputs are all validated, apart from address line 2, as this is an optional field.

* 1. **Admin panel- test the CRUD operation for a listing create, edit and delete function for a listing You can use the test plan template provided or create your own template using an industry-relevant format (use references for where you derived your template from).**

All listings and Enquiries sections had an array key error. This error was fixed by capitalizing “name” as in the database it is called “NAME”. Listing creation menu is simple to use, however while images can be selected on it, they do not show on the individual listing’s pages. The do however show on all listing’s pages. All other sections, including, enquires, all orders, manage users, and all listings, are very clear to use, and can be operated by non tech savvy people.

**Submission**

* If completely using Microsoft Word for all tasks, make it into a single file. If also using Microsoft Excel or any other files, then zip all files and upload them to Cloud Campus.
* Upload your updated website to Cloud Campus including the database.