|  |  |
| --- | --- |
| DLSU CCS black white.jpg | De La Salle University  College of Computer Studies  Secure Web Development Case Project Checklist |

|  |  |  |  |
| --- | --- | --- | --- |
| Name (LN, FN): |  | Date: |  |
| Section: |  | Grade: |  |

| **Requirement** | **Complete (2)** | **Incomplete (1)** | **Missing (0)** |
| --- | --- | --- | --- |
| 1. **Pre-demo Requirements (must be created before the actual demo)** |  |  |  |
| * 1. **Accounts (at least 1 per type of user)** |  |  |  |
| * + 1. Website Administrator |  |  |  |
| * + 1. Product Manager |  |  |  |
| * + 1. Customer |  |  |  |
| 1. **Demo Requirements** |  |  |  |
| * 1. **Authentication** |  |  |  |
| * + 1. Require authentication for all pages and resources, except those specifically intended to be public |  |  |  |
| * + 1. All authentication controls should fail securely |  |  |  |
| * + 1. Only cryptographically strong one-way salted hashes of passwords are stored |  |  |  |
| * + 1. Authentication failure responses should not indicate which part of the authentication data was incorrect. For example, instead of "Invalid username" or "Invalid password", just use "Invalid username and/or password" for both |  |  |  |
| * + 1. Enforce password complexity requirements established by policy or regulation |  |  |  |
| * + 1. Enforce password length requirements established by policy or regulation |  |  |  |
| * + 1. Password entry should be obscured on the user's screen (use of dots or asterisks on the display) |  |  |  |
| * + 1. Enforce account disabling after an established number of invalid login attempts (e.g., five   attempts is common). The account must be disabled for a period of time sufficient to discourage brute force guessing of credentials, but not so long as to allow for a denial-of-service attack to be performed |  |  |  |
| * + 1. Password reset questions should support sufficiently random answers. (e.g., "favorite book" is a bad question because “The Bible” is a very common answer) |  |  |  |
| * + 1. Prevent password re-use |  |  |  |
| * + 1. Passwords should be at least one day old before they can be changed, to prevent attacks on   password re-use |  |  |  |
| * + 1. The last use (successful or unsuccessful) of a user account should be reported to the user at their next successful login |  |  |  |
| * + 1. Re-authenticate users prior to performing critical operations such as password change |  |  |  |
| * 1. **Authorization/Access Control** |  |  |  |
| * + 1. Use a single site-wide component to check access authorization |  |  |  |
| * + 1. Access controls should fail securely |  |  |  |
| * + 1. Enforce application logic flows to comply with business rules |  |  |  |
| * 1. **Data Validation** |  |  |  |
| * + 1. All validation failures should result in input rejection. Sanitizing should not be used. |  |  |  |
| * + 1. Validate data range |  |  |  |
| * + 1. Validate data length |  |  |  |
| * 1. **Error Handling and Logging** |  |  |  |
| * + 1. Use error handlers that do not display debugging or stack trace information |  |  |  |
| * + 1. Implement generic error messages and use custom error pages |  |  |  |
| * + 1. Logging controls should support both success and failure of specified security events |  |  |  |
| * + 1. Restrict access to logs to only website administrators |  |  |  |
| * + 1. Log all input validation failures |  |  |  |
| * + 1. Log all authentication attempts, especially failures |  |  |  |
| * + 1. Log all access control failures |  |  |  |
| TOTAL |  | | |