Risk ID	Technical Risk	Technical Risk Indicators	Related CVE, CWE or OSVDB IDs	Impact Rating	Impact	Mitigation	Validation Steps
1	Code injection via dynamically evaluated untrusted user input.	Execution of eval() function on unstrusted user input.	CWE ID 98	H	Data loss, data corruption, stolen data, viruses, worms, denial of access, compromised machine	Validate user input-- ensure it adheres to correct form by validating data and sanitizing inputs. Avoid executing code derived from user input whenever possible.	Reject incorrectly formed data, only evaluate santizied input.
2	Code injection via the use of include() on untrusted user input	Execution of include() on untrusted user input.	CWE ID 98	H	Data loss, data corruption, stolen data, viruses, worms, denial of access, compromised machine	Validate user input-- ensure it adheres to correct form by validating data and sanitizing inputs. Avoid executing code derived from user input whenever possible.	Reject incorrectly formed data, only evaluate santizied input.
3	Dynamically constructed SQL query using untested user-supplied input for login (SQL Injection)	Execution of dynamically constructed mysql query for authentication on user input without any validation.	CWE ID 89	H	Data loss, data corruption, stolen data, denial of access, compromised account	Do not dynamically construct SQL queries using unvalidated user input. Validate and sanitize user input for correctnes and use parameterized pre-constructed queries.	Reject incorrectly formed data, only evaluate santizied input.

4	Dynamically constructed SQL query using untested user-supplied input for database access (SQL Injection)	Execution of dynamically constructed mysql query for data access on user input without any validation.	CWE ID 89	H	Data loss, data corruption, stolen data	Do not dynamically construct SQL queries using unvalidated user input. Validate and sanitize user input for correctnes and use parameterized pre-constructed queries.	Reject incorrectly formed data, only evaluate santizied input.
5	Password is hardcoded into code.	PHP variables called myUserName and myPassword	CWE ID 259	M	Compromised/stolen account, stolen data, impersonation, data corruption	Do not hardcode passwords into source code files. Follow best practices for protecting login credentials.	Remove all hardcoded passwords.
6	XSS Vulnerability	Dynamically constructed HTML elements from unsanitized user inputs.	CWE ID 80	M	Page data corruption, session stealing, stolen data available through HTML5 APIs, HTTP request manipulation	Sanitize input for script tags.	Sanitize all input for script tags.
7	Unchecked directory traversal	Filename is constructed using unvalidated user input.	CWE 73	M	Unauthorized data access	Check user input for proper form (prevent changing into any unecessary directory).	Reject user input that requests for data outside of authorized directories.
8	Information Leakage in Error Messages	Database error messages display unecessary sensitive information.	CWE 209	L	Sensitive information leaked	Remove unecessary data from error messages.	Don't show unecessary private data in error messages.

9	Unnecessary open port.	FTP service implemented in code.	CWE 220	M	Unauthorized data access, compromised root access	Close any unnecessary open ports. If some ports must stay open, ensure you have strong passwords and software with no known vulnerabilities.	Perform a scan for any open ports. Ensure only required ports come up.