Secure the code for this web application using OWASP!

Use the pseudocode for each file in the Github activity to make suggestions to secure the code. You can create a checklist on this form to achieve official OWASP hero status!

First name

Samson

Last name

Adeyemi

Login

-Input Validation:Limiting no of input prevent both XSS and SQL injection
-Parameterizing Variables: Also guide against SQL injection
- Use of Object Relational Mapper (ORM)
-The use of OWASP Enterprise Security API (ESAPI)must be enforce.
-Setting the cookies to HttpOnly prevent both XSS and SQL

Cookies

-Session ID should be 64-bit entropy & should not be in chronological No
-Limit the lenght of Session ID
-Session ID should not be save in the Database
-Hard code restriction on Session ID
-Session ID and Login credentials should not be kept in the cookies

XML Entities

-Disable external XML entities
-Do a validation checks
-secure the XML parsing APIs

Requests (GET/POST)

-Use GET for retrieving information.
-Use POST for information that will be manipulated or Updated.
-All POST requests should use HTTPS/SSL to ensure the body is encrypted (if you absolutely have to use HTTP, use it with GET).
-Vet any third party modules you use for creating GET/POST requests and use HTTPS for all of them!

URL Generation

-Restrict URL access with an authentication check
-Keep Session IDs out of URL
-Ensure that the database variables is not showing in the URL
-Instead of naming your target pages with meaning, use an array of key-value pairs that reference your objects.

SQL Queries

-For dynamic queries, use the sp_executesql function to keep them safe
-Use stored SQL procedures
-Use Object Relational Mapping

Database Encryption

-Use database masking
-Encrypt passwords and PII in database
-Access rights for DBAS should be limited
-Use anonymization, Pseudonymisation or data minimization to protect PII
-Use strong hash algorithms to encrypt your database such as Argon5,

Framework

-Document all APIs, languages, libraries in your framework
-Document security patching for vulnerabilities on all documented entities in your framework
-Ensure that patching is compatible with the rest of your framework with testing

Environment

-Ensure Physical Servers are sucured
-Secure configuration on firewalls
-Secure credentials for administrative logins on platform and applications
-Removal of default credentials