Extra Web Security Hardenings in PHP

1. Toolkit will Scan for the following

1. Making sure .sql .txt .zip .rar .tar .gz composer.json package.json bower.js files not stored inside the project so it won't be uploaded on the server mistakenly, then hackers may download them which leads to information leakage.

2. The following code can be dangerous:

```php
$redirect_url = $_GET['url'];
header("Location: " . $redirect_url);
```

   if user configures his browser to ignore the redirect, he may be able to access the rest of the page, so add `exit(); or die();` after `header();` function.

2. Some Verbal Advices

1. Make sure CSS files are different for users and for admins, because they might lead to information leakage which would reveal project's features or admin pages or privileges.
   https://cheatsheetseries.owasp.org/cheatsheets/Securing_Cascading_Style_Sheets_Cheat_Sheet.html

2. Never put sensitive info in HTML, CSS or Javascript Comments.

```
<!-- NEVER PUT SENSITIVE DATA HERE -->
// NEVER PUT SENSITIVE DATA HERE
/* NEVER PUT SENSITIVE DATA HERE */
```

3. Make sure to minify and uglify HTML, CSS and Javascript Codes in production.

4. Use TLS/HTTPS and HSTS (maybe from cloudflare.com (its free)), Enable Force HTTPS.

5. Also cloudflare protects against DDOS attacks, it caches and minifies the whole client-side files which increases performance.

6. You can implement your own Denial of Service Prevention Mechanism, by logging number of requests incoming from specific IP, for example, if more than 100 Requests came from the same IP address per 1 Minute, this is an abnormal behavior, and the IP should be blocked for example (10 Minutes or More).

7. Logging And Monitoring
   i. Do logging of important situations such as login and logout and while error happens
   ii. Inner error reporting tool, run technical error reporting function in critical places in your code, where errors shouldn't be happening or user shouldn't be there etc..
   iii. Limit logging per sometime so logging system does not overflow, if DDoS attack occurred.

8. Checking admin table in the database
   i. The real legit admin account shouldn't have id number 1, insert dummy inactive record to the first row of admin table with id 1, so if admin table's info somehow gets leaked, it won't show the first admin row credentials.