Отчет: аудит безопасности

Задание 7

Уязвимости:

1. Cross Site Scripting (XSS)	**Фильтрация (preg_match)** Она определяет данные, вводимые пользователем на соответствие заданным требованиям. **Приведение к типу (is_numeric)** Проверяет, является ли переменная числом или же числовой строкой.
2. SQL Injection	Использование подготовленных запросов (PDO), что позволяет отделить данные пользователя от данных запроса. Экранирование(**addslashes**) Возвращает строку с обратным слешом перед символами, которые нужно экранировать. Экранируются одиночная кавычка ('), двойная кавычка ("), обратный слеш (\) и NUL (NULL байт).
3. Cross Site Request Forgery (CSRF)	Использование методов POST и GET по назначению. XNR токены.
4. Upload и Include	Использовался полный путь к файлу, чтобы избавиться от вероятности, что файл будет подменен (**dirname**)

Примеры из кода:

1. XSS:

```
if (empty($_POST['fio']) || preg_match("/^[А-ЯЁ][а-яё]*$/", $_POST['fio']))
```

```
if (empty($_POST['year']) || !is_numeric($_POST['year']) || !preg_match('/^\d+$/', $_POST['year']))
```

2. SQL Injection

```
$user = 'u52858';
$pass = '6454527';
$db = new PDO('mysql:host=localhost;dbname=u52858', $user, $pass,
  [PDO::ATTR_PERSISTENT => true, PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION]);


try {
  $stmt = $db->prepare("INSERT INTO req (name, year, email, gender, limbs, biography) VALUES (:name, :year, :email, :gender, :limbs, :biography)");
  $stmt->bindParam(':name', $_POST['fio']);
  $stmt->bindParam(':year', $_POST['year']);
  $stmt->bindParam(':email', $_POST['email']);
  $stmt->bindParam(':gender', $_POST['gender']);
  $stmt->bindParam(':limbs', $_POST['limbs']);
  $stmt->bindParam(':biography', $_POST['biography']);
  echo('Запись1<br/>');
  $stmt->execute();
```

```
$stmt3->bindParam(':name', addslashes($_POST['fio']));
$stmt3->bindParam(':year', $_POST['year']);
$stmt3->bindParam(':email', addslashes($_POST['email']));
$stmt3->bindParam(':gender', $_POST['gender']);
$stmt3->bindParam(':limbs', $_POST['limbs']);
$stmt3->bindParam(':biography', addslashes($_POST['biography']));
$stmt3->execute();
```

3. CSRF

```php
// Созадние токена
$token = bin2hex(random_bytes(32));
$time=time();
$user = 'u52882'; $pass = '8244733';
$db = new PDO('mysql:host=localhost;dbname=u52882', $user, $pass, [PDO::ATTR_PERSISTENT => true]);
$statement = $db->prepare("REPLACE INTO Tokens(token, timetok) VALUES (:token, :timetok)");
$rest=$statement->execute(['token'=>"$token",'timetok'=>"$time"]);
```

```php
// Проверка работы токена
$token = $_POST['token'];
$time = $_POST["timetok"];
$user = 'u52882'; $pass = '8244733';
$db = new PDO('mysql:host=localhost;dbname=u52879', $user, $pass, [PDO::ATTR_PERSISTENT => true]);
$statement = $db->prepare("SELECT * FROM Tokens where token=:token");
$statement->execute(['token'=>"$token"]);
$rt = $statement->fetchAll();
if($rt[0]['token']!=$_POST['token'] || $rt[0]['timetok']-time()>60*60){
    echo "Произошла ошибка! Попробуйте снова..";
    exit();
}
```

4. Upload и Include

```php
include(dirname(__FILE__).'form.php');

// Иначе, если запрос был методом POST, т.е. нужно проверить
```