**Защита от XSS**

    function checkInput($param){

        return htmlspecialchars(strip\_tags(trim($param)), ENT\_QUOTES);

    }

Пример использования:

    $fio = (isset($\_POST['fio']) ? checkInput($\_POST['fio']) : '');

Эта функция используется при получении данных из вне, чтобы предотвратить внедрение js скриптов. Она удаляет лишние пробелы в начале и в конце, удаляет html тэги и преобразует в html объекты символы.

**Защита от Information Disclosure**

Каждому пользователю выделена своя роль. У админа – admin, у пользователя – user. При открытии панели выполняется проверка:

  if(isset($\_SERVER['PHP\_AUTH\_USER']) && isset($\_SERVER['PHP\_AUTH\_PW'])){

    $qu = $db->prepare("SELECT id FROM users WHERE role = 'admin' and login = ? and password = ?");

    $qu->execute([$\_SERVER['PHP\_AUTH\_USER'], md5($\_SERVER['PHP\_AUTH\_PW'])]);

    $haveAdmin = $qu->rowCount();

  }

Так же при открытии страницы формы, проверяется на то, что пользователь авторизован или нет. Если не авторизован, выводится пустая форма, иначе по базе ищет данные формы с этим пользователем. Если о него не найдет, выполнится функция:

  function user\_exit(){

    del\_cook\_all(1);

    session\_destroy();

    header('Location: index.php');

    exit();

  }

Эта функция выполняет logout пользователя удаляет все кукисы и выводит пустую форму.

**Защита от SQL Injection**

Используется PDO (подготовленные запросы) и дополнительно вызывается функция checkInput (выше) для дополнительной защиты.

        $stmt = $db->prepare("INSERT INTO users (login, password) VALUES (?, ?)");

        $stmt->execute([$login, $mpassword]);

**Защита от CSRF**

Проверяется совпадение подлинности запроса с данными, которые хранятся в сессии

Генерируем csrf\_token, сохраняем в сессию и выводим в форму

<?php

$csrf\_token = bin2hex(random\_bytes(32));

$\_SESSION['csrf\_token'] = $csrf\_token;

?>

<input type="hidden" name='csrf\_token' value='<?php echo $csrf\_token; ?>'>

При поступлении запроса на сервер, получаем из сессии и из POST csrf\_token и сравниваем

$csrf\_error = (isset($\_COOKIE['csrf\_error']) ? checkInput($\_COOKIE['csrf\_error']) : '');

if (isset($\_COOKIE['csrf\_error'])) {

$messages['error'] = 'Не соответствие CSRF токена';

      del\_cook('csrf');

}

Данная проверка используется при отправки формы, при авторизации, при отправки запросов из панели администратора.

**Защита от Include уязвимости**

В php не подключаются сторонние файлы, по этому в моем коде данная проверка не реализована.

Если бы было подключение, нужно было использовать белый список разрешенных файлов, который в процессе выполнения кода проверялась

**Защита от Upload уязвимости**

Так как в форме обратной связи не предусмотрена загрузка файлов, данная проверка не реализована.

Если бы загружались файлы, то нужно было проверить тип файла, его размер (ограничить максимальный размер на стороне сервера) и запретить выполнение скриптов в каталоге, где находились бы загруженные файлы.