Защита от Include уязвимости.

- Использование абсолютного пути для включения файла db_pass.php с помощью __DIR__. А также, переменные инициализируются в начале скрипта для предотвращения неинициализированных ошибок.

```php
<?php
session_start();

include __DIR__ . '/db_pass.php';

$expiration_time = time() + 365 * 24 * 60 * 60;

function test_input($data)
{
  ...
}

$fullname = "";
$email = "";
$phone = "";
$bio = "";
$gender = "";
$contact = "";
$selected_languages = [];
```

Защита от XSS

- Для предотвращения выполнения вредоносного кода, который может быть внедрен в базу данных, все данные, которые выводятся в HTML, а также все сообщения об ошибках выводятся через функцию htmlspecialchars().

  Данный способ используется в таких файлах, как adminPage.php, delete_user.php, form.php, formhandler.php, login.php, user_profile.php.

  Пример использования из adminPage.php (строки 39-53) :

```php
        <tbody>
            <?php foreach ($users as $user): ?>
                <tr>
                    <td><?= htmlspecialchars($user['user_id']) ?></td>
                    <td><?= htmlspecialchars($user['fullname']) ?></td>
                    <td><?= htmlspecialchars($user['phone']) ?></td>
                    <td><?= htmlspecialchars($user['email']) ?></td>
                    <td><?= htmlspecialchars($user['dob']) ?></td>
                    <td><?= htmlspecialchars($user['gender']) ?></td>
```

```
                    <td><?= htmlspecialchars($user['bio']) ?></td>
                    <td><a href="user_profile.php?user_id=<?=
htmlspecialchars($user['user_id']) ?>">Редактировать</a></td>
                    <td><a href="delete_user.php?user_id=<?=
htmlspecialchars($user['user_id']) ?>">Удалить</a></td>
                </tr>
            <?php endforeach; ?>
        </tbody>
```

- Функция test_input(), созданная для обработки входных данных. Имеется две разных версии, подстроенные под данные из файлов.

  Версия в formhandler.php (строчки 8-14):

```
function test_input($data) {
    $data = trim($data);
    $data = stripslashes($data);
    $data = strip_tags($data);
    $data = htmlspecialchars($data);
    return $data;
}
```

  Версия в user_profile.php (строчки 12-18):

```
        function test_input($data)
        {
            $data = trim($data);
            $data = stripslashes($data);
            $data = htmlspecialchars($data);
            return $data;
        }
```

- В formhandler.php при генерации пароля используется функция password_hash() для его хеширования и хранения в виде хеш-текста. Это обеспечивает дополнительный слой безопасности от утечки данных пользователей.

  (строчка 115):

```
$generated_password = password_hash($password, PASSWORD_DEFAULT);
```

- В login.php используется функция password_verify() вместо прямого сравнения для чтения их в виде хеш-текстов и их сохранности.

  (строчка 14):

```
if ($user && password_verify($password, $user['password']))
{
   ...
}
```

Защита от CSRF

- Создает уникальный токен csrf_token для валидации данных, хранящихся в активной сессии.

В файле form.php (строки 4-8):

```
if (!isset($_SESSION['csrf_token']))
    {
        $_SESSION['csrf_token'] = bin2hex(random_bytes(32));
    }
    $token = $_SESSION['csrf_token'];
```

В том же файле, в начале html части (строка 80):

```
<body class="form-body">
    <div class="form-container">
        <h2>Форма</h2>
        <form method="POST" id="myForm" action="formhandler.php">
        <input type="hidden" name="token" value="<?= $token; ?>">
```

Защита от SQL Injection

- Валидация полей с помощью регулярного выражения loginReq и passReq позволяет обеспечить защиту против вредоносных командных запросов, введенных в поля для входа.

В файле login.php (строки 4-20):

```
    $loginReq = "/^[a-zA-Z]+_[0-9]+$/";
    $passReq = '/^[a-zA-Z0-9]+$/';

    if ($_SERVER["REQUEST_METHOD"] == "POST") {
        if(isset($_POST['enterAcc'])) {
            $login = $_POST['login'];
            if(!preg_match($loginReq,$login))
            {
                echo "Incorrect Username";
                exit();
            }
            $password = $_POST['password'];
            if(!preg_match($passReq,$password))
            {
```

```
                echo "Incorrect Password";
                exit();
            }
```

Защита от Information Disclosure

- Защита от раскрытия информация, реализованная за счёт запрета захода на страницу если пользователь не залогинен или не имеет права администратора.

  В файле adminPage.php (строки 5-22):

```
    if(empty($_SESSION['username']))
    {
        header("Location:index.php");
        exit();
    }

    if(!empty($_SESSION['username']))
    {
        $username = $_SESSION['username'];
        $stmt = $db->prepare("SELECT * FROM Users WHERE username = ? and admin
= 1");
        $stmt->execute([$username]);
        $userAdmin = $stmt->fetch(PDO::FETCH_ASSOC);
        if(empty($userAdmin))
        {
            header("Location:index.php");
            exit();
        }
    }
```