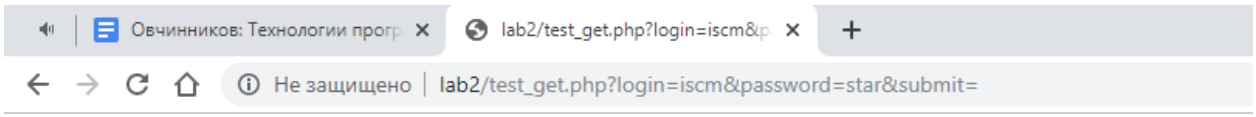
Use case

- Имеется форма, в которой 2 поля: логин и пароль и кнопка «Войти»
- Поле «Логин» имеет тип text, поле «пароль» тип password, кнопка тип submit
- Страница ошибки представляет собой «шаблон» странички, куда отправляется переменная с текстом ошибки в зависимости от характера ошибки: неправильный логин, неправильный пароль
- После авторизации происходит создание сессии и пользователя направляют на страницу с секретной информацией, в данном случае это профиль странички для примера.
- На странице профиля выводится личная информация из бд о человеке, также выводятся две таблички с данными о детьми и супруге. Данные о детей и супруга хранятся в разных таблицах в бд, они связаны с таблицей личной информации человека.
- Кнопка «Выход» находится в меню. При нажатии удаляются все данные сессии и перенаправляет пользователя на главную страницу.

Контрольные вопросы:

1. Какой тип запроса к веб-серверу следует использовать для отправки данных авторизации?

Метод POST не предоставляет информацию через URL-адрес, он предоставляет столько же информации, сколько GET в фактической сетевой связи между клиентом и сервером. По протоколу HTTPS данные POST кодируются.
Для примера был создан файл “test_get.php” куда отправлялись данные с использованием метода GET. Данные легкодоступны



2. Что такое хэш-функция? Для чего она нужна в механизме авторизации.

Хэш-функция – это односторонняя функция шифрования необходимая для получения из сообщения произвольного размера его дайджеста – значения фиксированного размера. Дайджест может быть использован в качестве контрольной суммы исходного сообщения, обеспечивая таким образом (при использовании соответствующего протокола) контроль целостности информации. Основные свойства хэш-функции:

- на вход хэш-функции подается сообщение произвольной длины;
- на выходе хэш-функции формируется блок данных фиксированной длины;
- значения на выходе хэш-функции распределены по равномерному закону;
- при изменении одного бита на входе хэш-функции существенно изменяется выход.

Хэш-функции необходимы для защиты данных пользователя.
В моем проекте уместное место для применения хеш-функции является после ввода информации и проверки на корректность, в обработке данных полученных от пользователя.

3. Какие “уязвимости” в созданном механизме авторизации вы видите?

С помощью прямой ссылки можно было бы получить секретную информацию, перейдя на страницу с информацией по прямой ссылке, но проверка на наличие данных в сессии в начале файла поможет избежать этого. В случае не авторизованного пользователя, его направят на страницу входа:

```
<?php
session_start();
if(!(isset($_SESSION['username'])))
{
    header("Location: entry.php");
    exit;
}
```

В cookie можно хранить лишь идентификатор пользователя. Если сохранять не только идентификатор в cookie, то можно изменять его как угодно и войти как другой пользователь. В обычно хранят $_SESSION[‘id’]. Персональные данные недопустимо хранить.

4. Как сделать чтобы после авторизации обновление страницы не вызывало запрос браузера вида “требуется повторная отправка формы”?

После авторизации данные проходят проверку и если все в порядке, пользователя перенаправляют на страницу с секретной информацией:

```
if($password == $hash["$login"])
    {
      $_SESSION['username'] = $login;
      header("Location: profile.php");
    }
```

5. Чем отличается авторизация на профессионально сделанном сайте от того, что сделали вы?

В подавляющих случаях на сайтах детских садов отсутствует регистрация (если они есть, то я их не нашел). В основном есть форма для записи после отправки которых с родителями связываются по телефону, либо приглашают на встречу.
Я перешел на сайты 20 частных детских садов Москвы (лучшие по версии: https://chips-journal.ru/reviews/luchshikh-chastnykh-detskikh-sadov-moskvy) и не на одной не было регистрации.

Если сравнить с другими сайтами, то думаю, что отличия в функциональности и в том что все данные хранятся в базе данных в виде хэшей.