Лабораторная работа №1

Основное задание

Спроектировать и разработать систему авторизации пользователей на протоколе HTTP. Система должна обладать следующим функционалом:

- [x] Функциональность входа и выхода
- [x] Пароли должны хранится в хешированном виде

Дополнительные задания

В качестве дополнительного функционала можно реализовать следующие задачи:

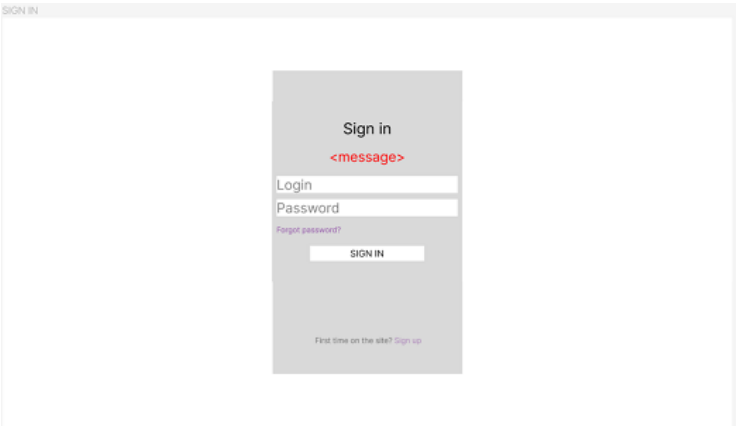
- [x] Форма регистрации
- [x] Возможность смены пароля
- [x] Возможность восстановления пароля
- [x] Ограничение времени сессии на стороне сервера
- [x] Ограничение срока действия пароля на стороне сервера
- [x] Хранение хеша пароля с солью
- [x] Возможность одновременного использования одним пользователем нескольких клиентов

Ход работы

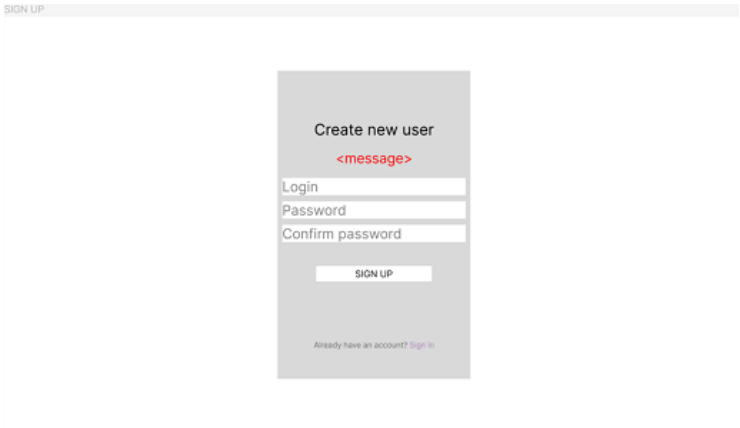
1. Разработка пользовательского интерфейса

В редакторе Figma был разработан пользовательский интерфейс

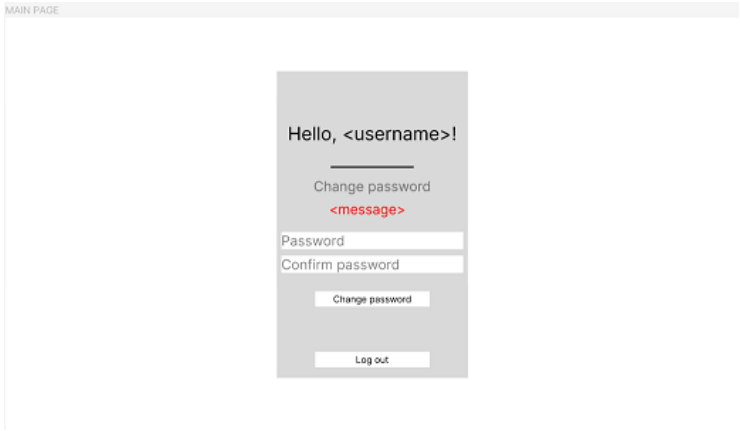
- Страница авторизации:



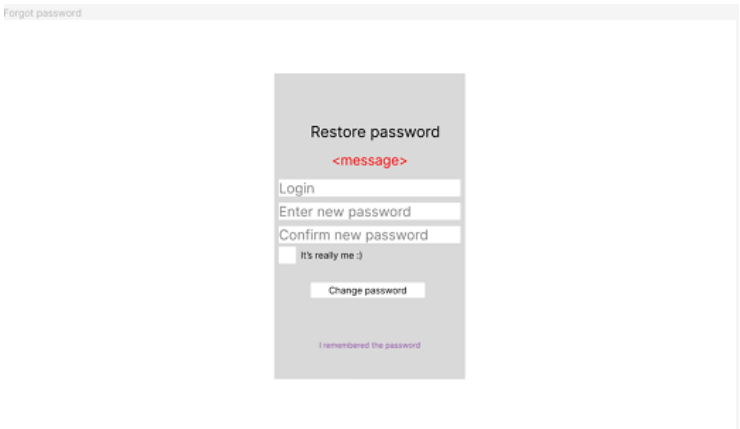
- Страница регистрации:



- Главная страница:



- Страница восстановления пароля:



2. Описание пользовательских сценариев работы

На сайте пользователю доступны следующие возможности:

- Регистрация
- Авторизация
- Восстановление пароля

При вводе неверных данных, пользователю выводится сообщение об ошибке.

После авторизации пользователю открываются следующие возможности:

- Смена пароля
- Выход из аккаунта

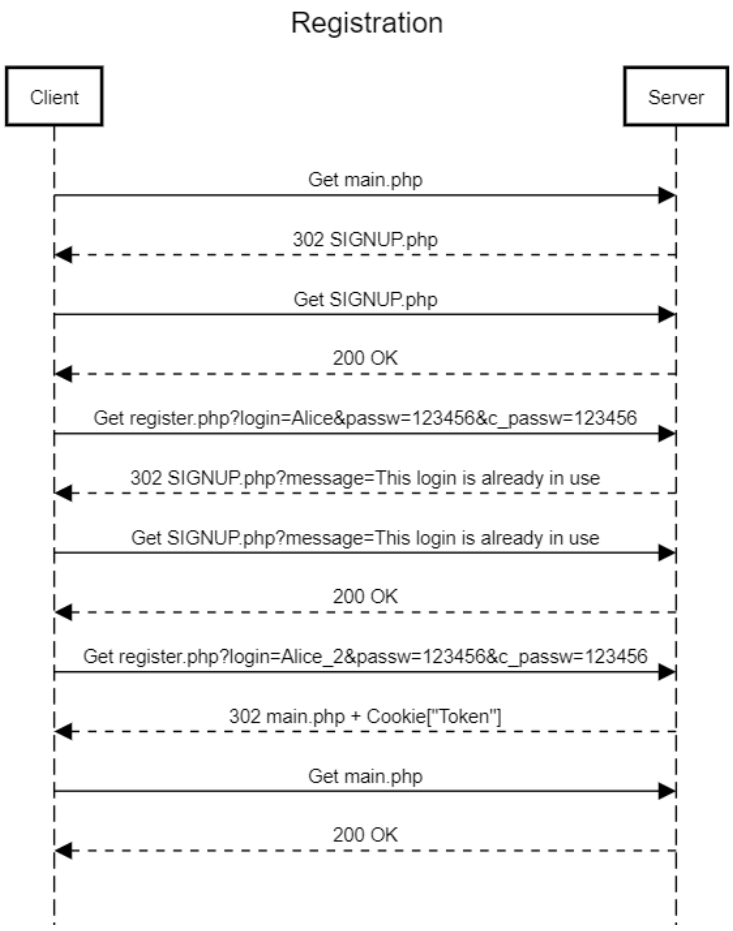
Если пользователь был ранее авторизован на сайте, у него есть cookie с токеном, и срок годности токена не истек, то при попытке зайти на страницы регистрации, авторизации и восстановления пароля, он будет автоматически перенаправлен на главную страницу.

После того как срок годности токена закончится, при обновлении главной страницы, будет произведен автоматический выход из аккаунта. Пользователь будет перенаправлен на главную страницу и ему будет выведено сообщение, что срок действия его сессии вышел.

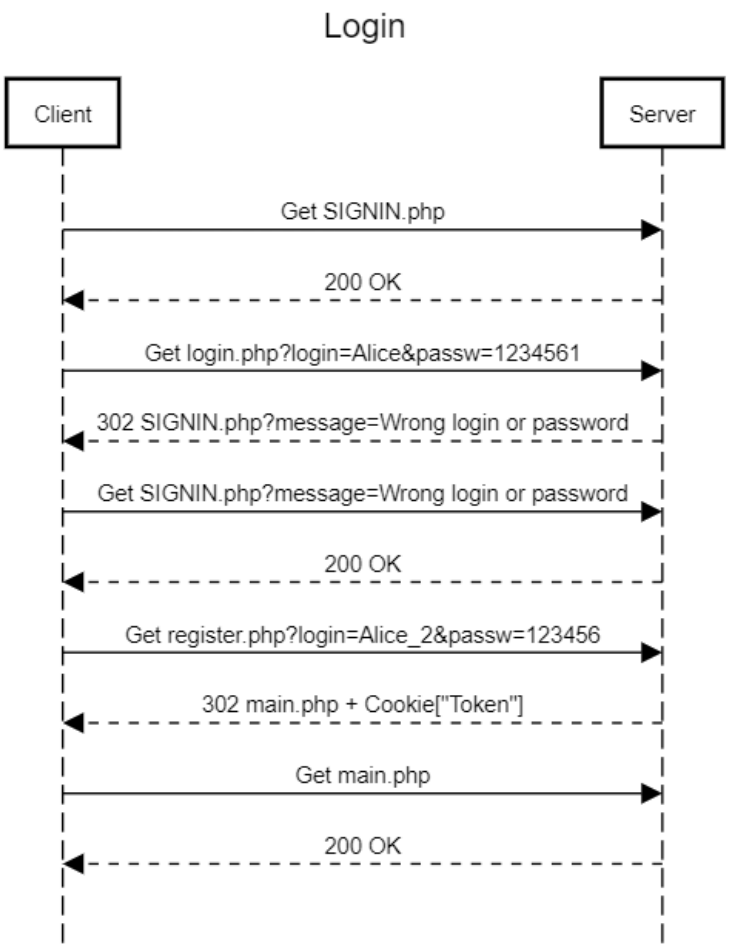
Если в течении недели пользователь ни разу не обновит свой пароль, при переходе на главную страницу ему будет выведено сообщение с предложением сменить пароль.

3. Описание API сервера и хореографии

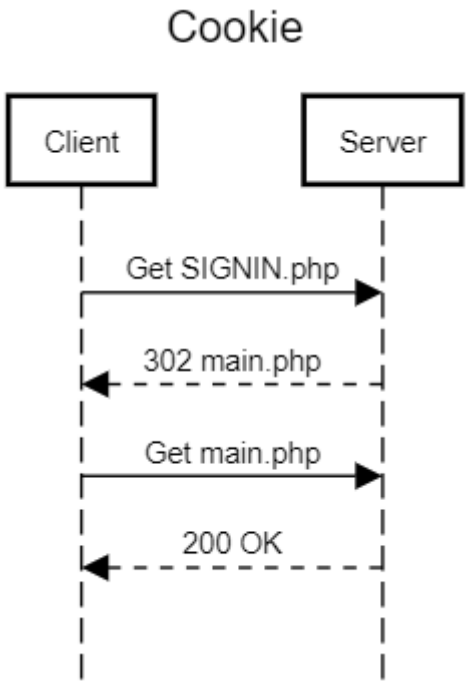
- Пример запросов, когда пользователь впервые заходит на страницу main.php, а после регистрируется:



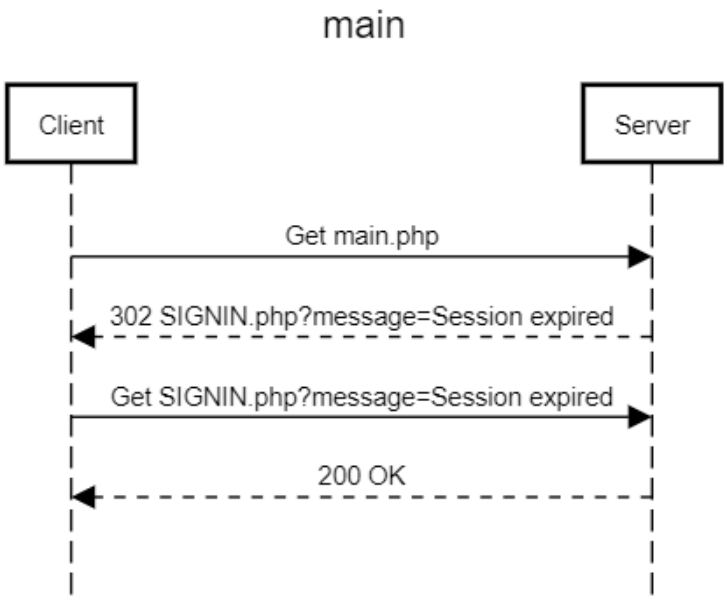
- Пример запросов, когда пользователь заходит на страницу авторизации, вводит неверное имя пользователя, а после вводит верные данные и входит в аккаунт:



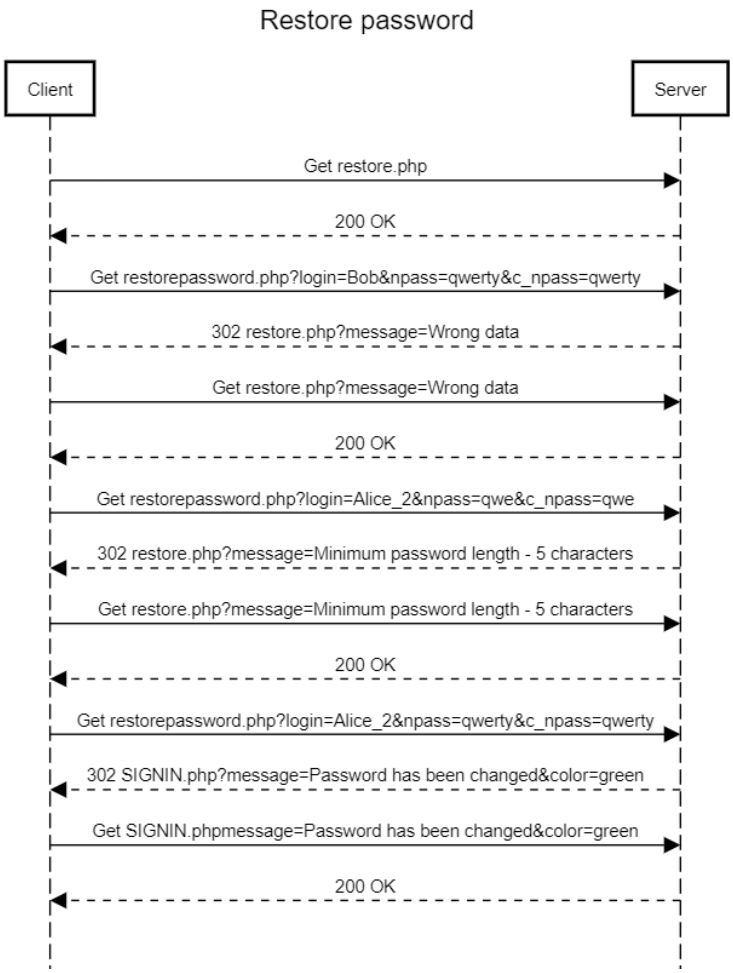
- Пример запросов, когда пользователь заходит на страницу авторизации и у него есть cookie с токеном:



- Пример запросов, когда пользователь заходит на главную страницу с просроченным токеном:



- Пример запросов, когда пользователь пытается восстановить пароль, вводя логин не существующего пользователя, а затем задавая слишком короткий пароль:



4. Описание структуры базы данных

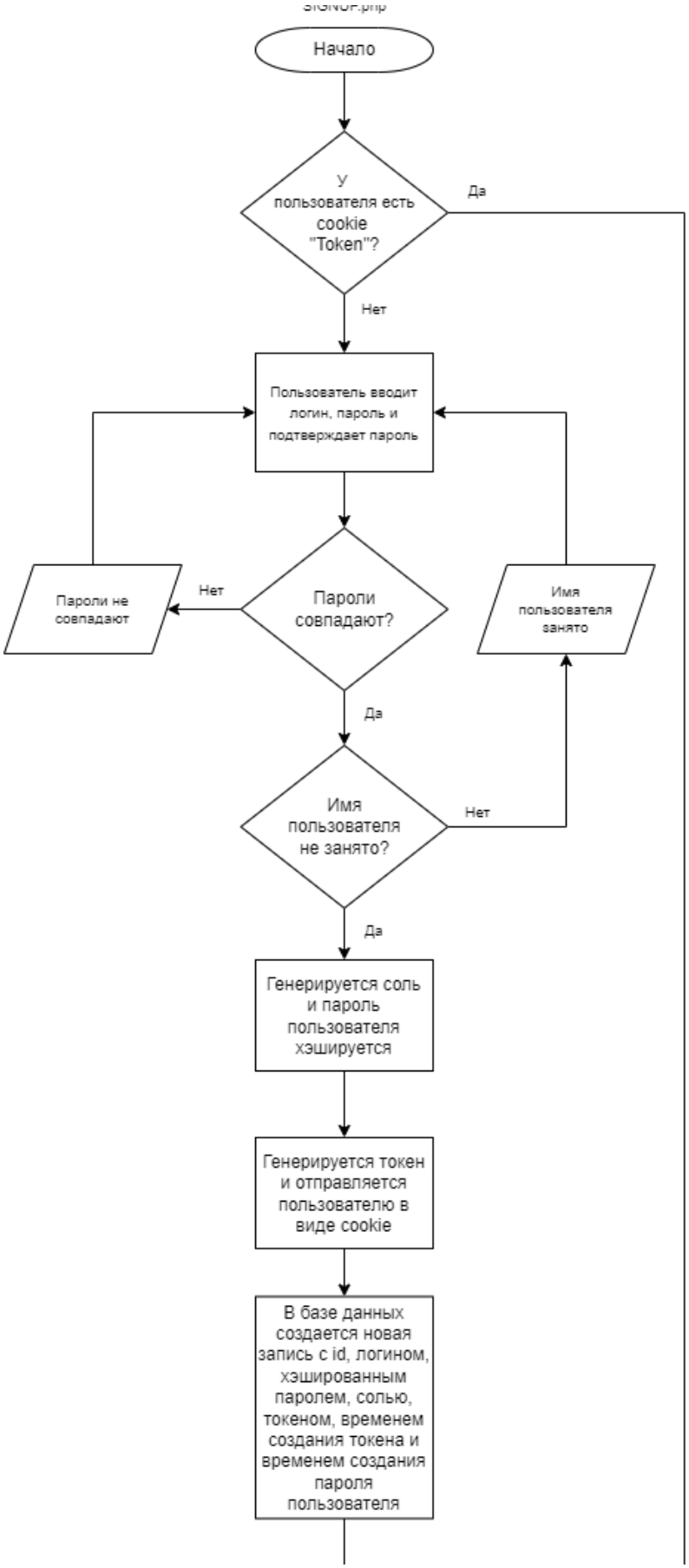
Для хранения данных пользователей используется JSON файл. Каждый объект содержит в себе индивидуальный номер пользователя, имя пользователя, хэш пароля, соль, токен, время создания токена и время создания пароля.

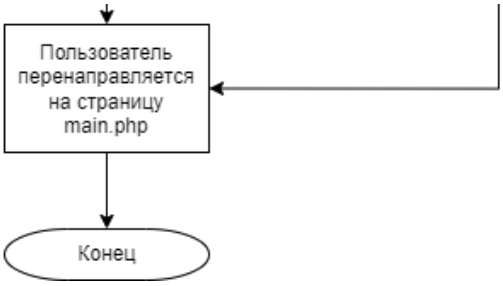
Пример данных пользователя в базе данных:

```
{
    "Id":3,
    "Login":"new_user",
    "Password":"b7387a2deb85d1ea99d3b74fcf92c6d3",
    "Salt":"k3ORiHLbYi",
    "Token":"XJNQNL27jYWP597Gtsok",
    "Time":1664946806,
    "Time_p":1664946816
}
```

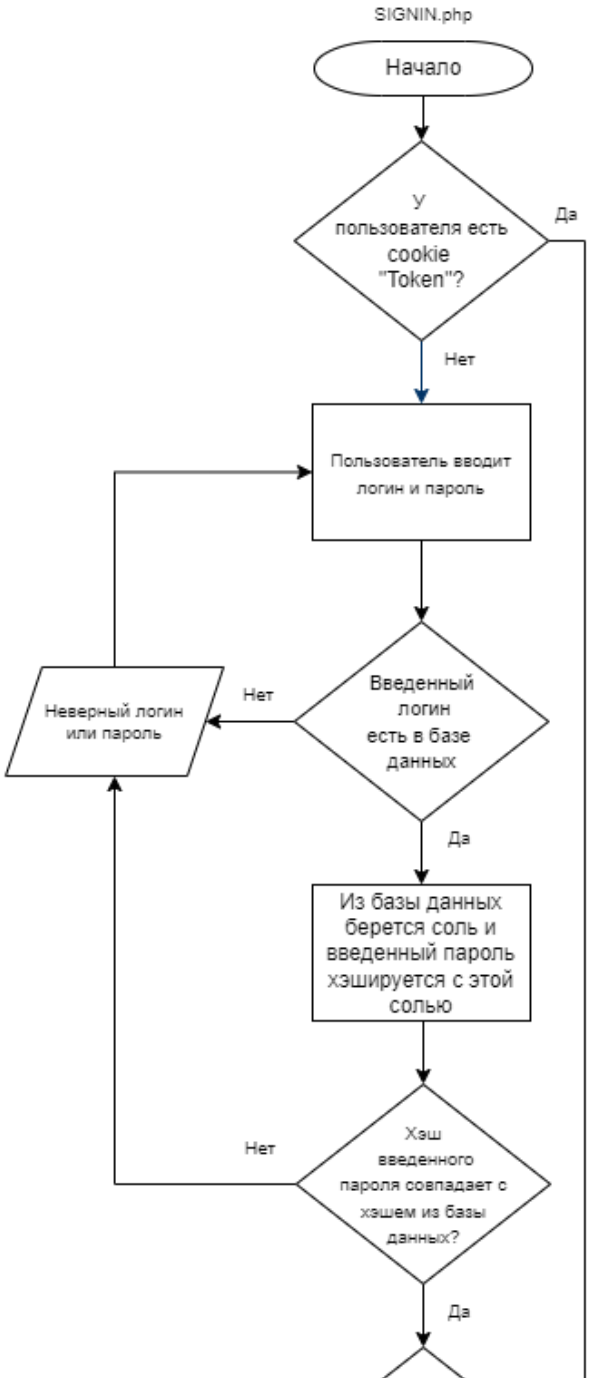
5. Описание алгоритмов

- Алгоритм страницы регистрации:

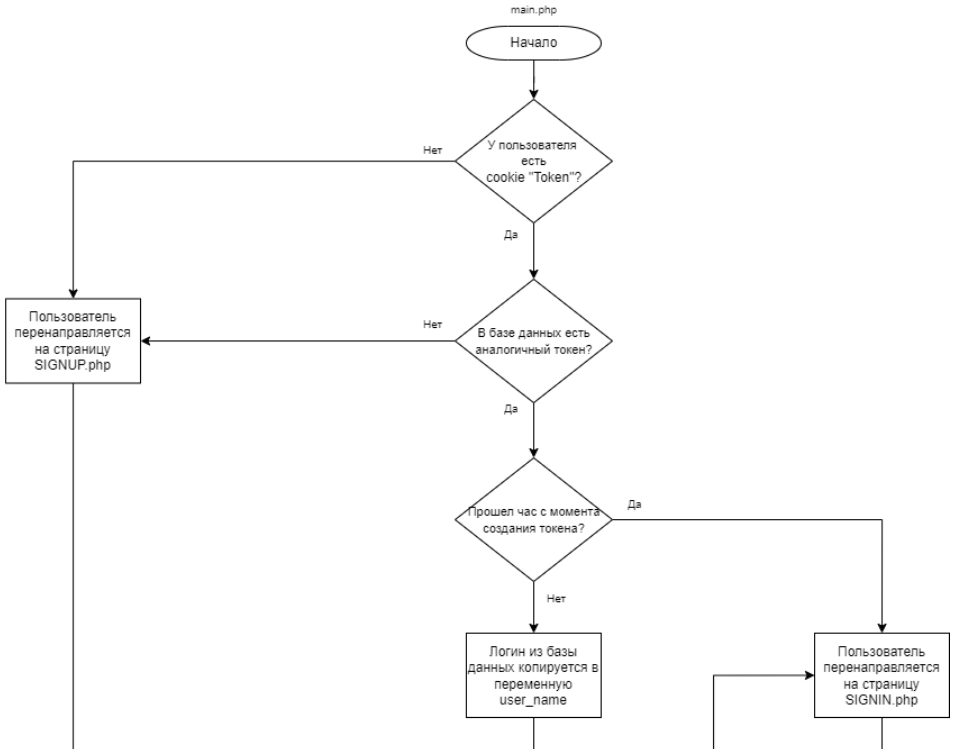

Начало
У пользователя есть cookie "Token"?
Да
Нет
Пользователь вводит логин, пароль и подтверждает пароль
Пароли совпадают?
Нет
Пароли не совпадают
Да
Имя пользователя не занято?
Нет
Имя пользователя занято
Да
Генерируется соль и пароль пользователя хэшируется
Генерируется токен и отправляется пользователю в виде cookie
В базе данных создается новая запись с id, логином, хэшированным паролем, солью, токеном, временем создания токена и временем создания пароля пользователя

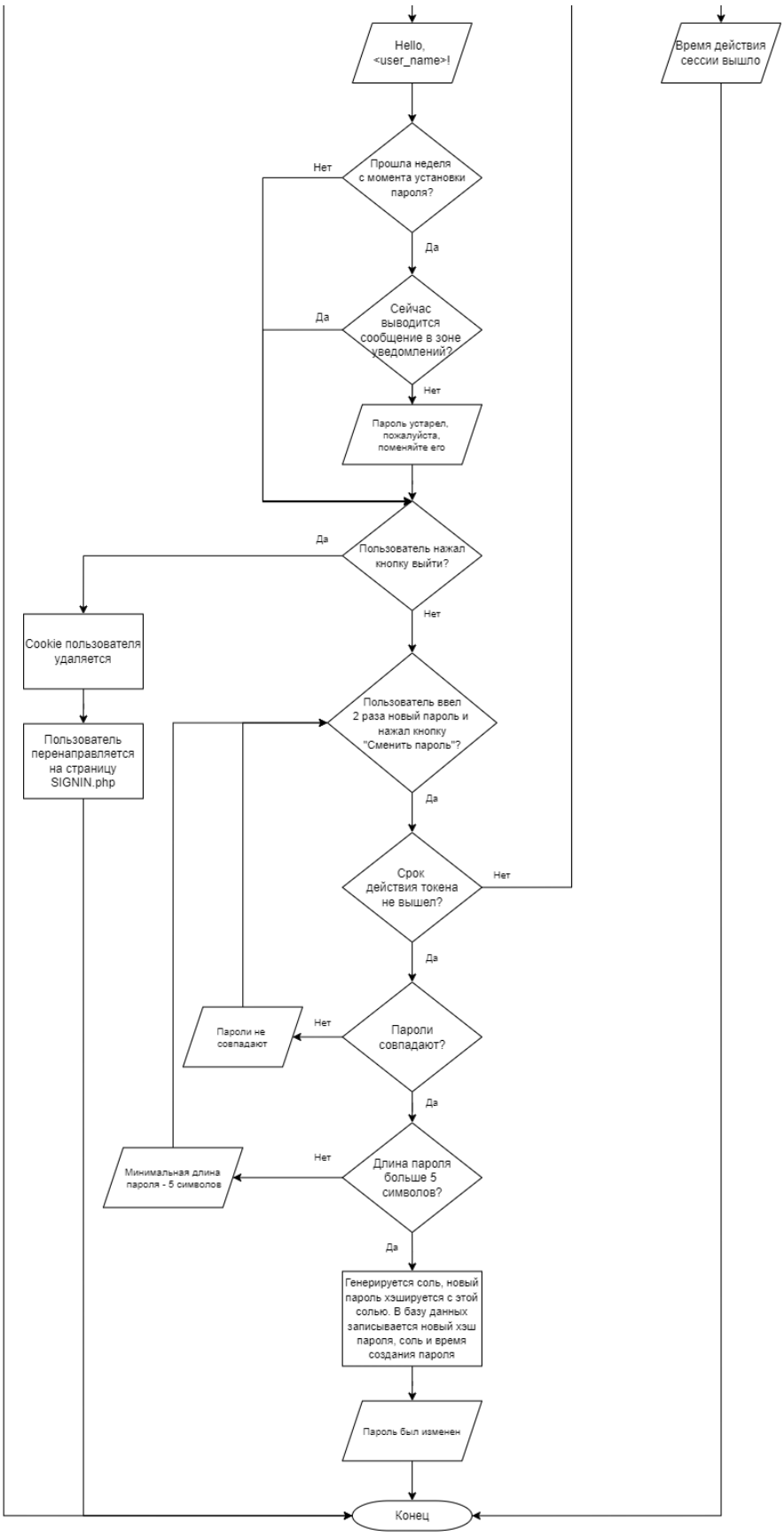


- Алгоритм страницы авторизации:

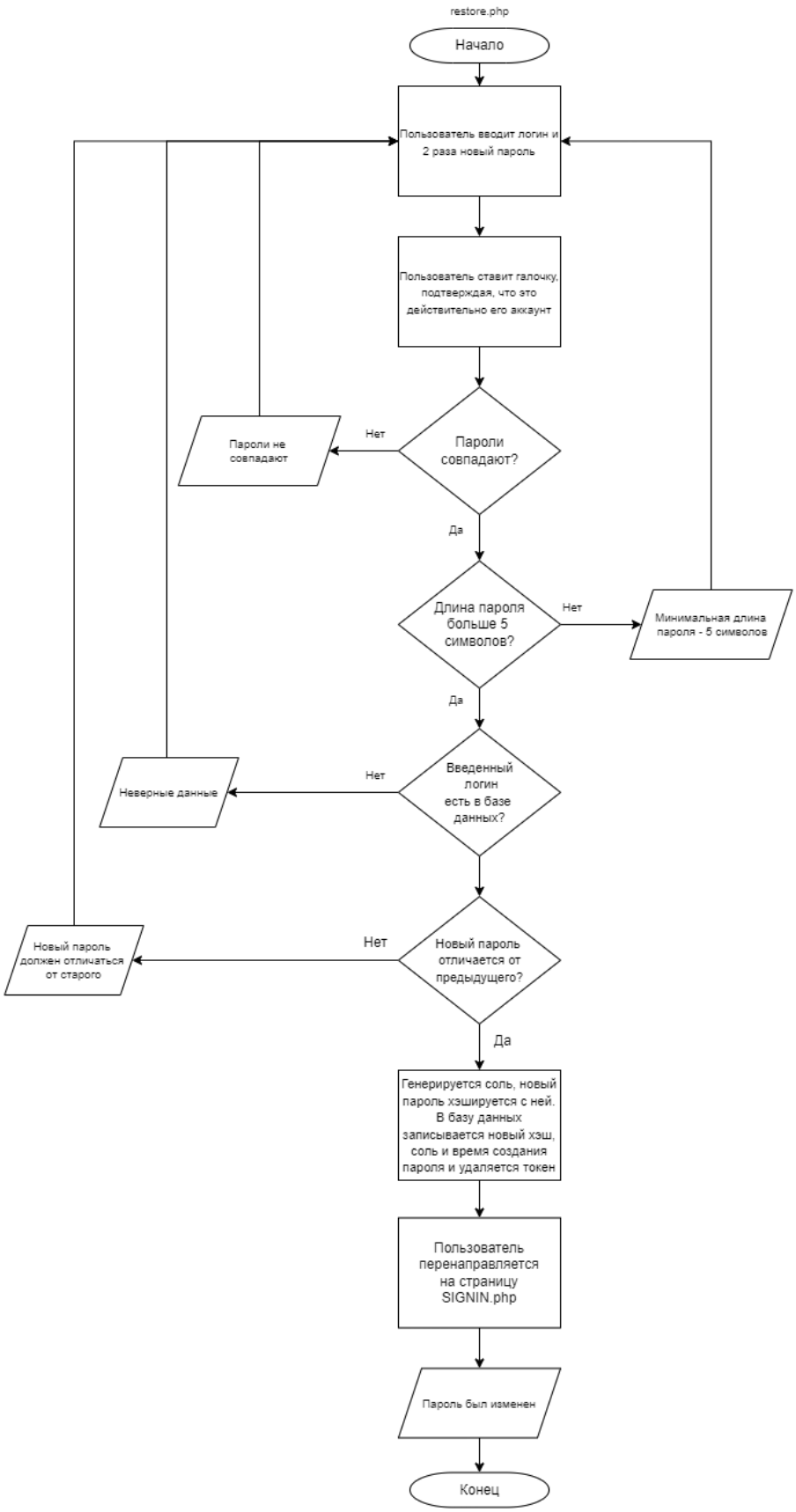


- Алгоритм главной страницы:




Hello, <user_name>!
Время действия сессии вышло
Прошла неделя с момента установки пароля?
Нет
Да
Сейчас выводится сообщение в зоне уведомлений?
Да
Нет
Пароль устарел, пожалуйста, поменяйте его
Пользователь нажал кнопку выйти?
Да
Нет
Cookie пользователя удаляется
Пользователь перенаправляется на страницу SIGNIN.php
Пользователь ввел 2 раза новый пароль и нажал кнопку "Сменить пароль"?
Да
Срок действия токена не вышел?
Нет
Да
Пароли не совпадают
Нет
Пароли совпадают?
Да
Минимальная длина пароля - 5 символов
Нет
Длина пароля больше 5 символов?
Да
Генерируется соль, новый пароль хэшируется с этой солью. В базу данных записывается новый хэш пароля, соль и время создания пароля
Пароль был изменен
Конец

- Алгоритм страницы восстановления пароля:


restore.php
Начало
Пользователь вводит логин и 2 раза новый пароль
Пользователь ставит галочку, подтверждая, что это действительно его аккаунт
Пароли совпадают?
Нет
Пароли не совпадают
Да
Длина пароля больше 5 символов?
Нет
Минимальная длина пароля - 5 символов
Да
Введенный логин есть в базе данных?
Нет
Неверные данные
Да
Новый пароль отличается от предыдущего?
Нет
Новый пароль должен отличаться от старого
Да
Генерируется соль, новый пароль хэшируется с ней. В базу данных записывается новый хэш, соль и время создания пароля и удаляется токен
Пользователь перенаправляется на страницу SIGNIN.php
Пароль был изменен
Конец

Значимые фрагменты кода

Фрагмент кода проверки данных, полученных из формы регистрации:

```
check_passwords($password, $c_password, "http://localhost/SIGNUP.php");
valid_password($password, "http://localhost/SIGNUP.php");

$db = open_data("data.json");

foreach($db as $ac) {
    if ($ac["Login"] == $user) {
        header("Location: http://localhost/SIGNUP.php?message=This login is already in
use");
        exit;
    }
}
```

Фрагмент кода проверки введенных данных и обновления токена при авторизации:

```
foreach($db as &$ac) {
        if ($user == $ac["Login"]) {
            [$h_password, $h_salt] = hash_password($password, $ac["Salt"]);

            if ($h_password == $ac["Password"]) {
                if (($ac["Token"] != null) && ($ac["Time"] + 3600 > time())) {
                    $ac["Time"] = time();
                    setcookie("Token", $ac["Token"]);
                    file_put_contents("data.json", json_encode($db));
                }
                else {
                    $token = gen_str(20);
                    $ac["Token"] = $token;
                    $ac["Time"] = time();
                    setcookie("Token", $token);
                    file_put_contents("data.json", json_encode($db));
                }

                header("Location: http://localhost/main.php");
                exit;
            }
        }
    }
```

Функция проверки токена на наличие в базе данных и его срока действия:

```
function check_token($db) {
    if (isset($_COOKIE["Token"])) {
        $token = $_COOKIE["Token"];
        foreach($db as &$ac) {
            if ($ac["Token"] == $token) {
                if ($ac["Time"] + 3600 < time()) {
                    $ac["Token"] = null;
```

```
                    setcookie("Token", "", time() - 3600);
                    file_put_contents("data.json", json_encode($db));

                    header("Location: http://localhost/SIGNIN.php?message=Session
expired");
                    exit;
                }
                else    return $ac["Login"];
            }
        }

        header("Location: http://localhost/SIGNUP.php");
        setcookie("Token", "", time() - 3600);
        exit;
    }
}
```

Функция хэширования пароля:

```
function hash_password($password, $h_salt = null) {
    $h_salt = is_null($h_salt) ? gen_str(10) : $h_salt;
    $password = $password.$h_salt;
    $hash = hash("md5", $password);

    return array($hash, $h_salt);
}
```

Функция вывода сообщений на html страницах:

```
function print_message() {
    if (isset($_GET["message"])) {

        $color = isset($_GET["color"]) ? $_GET["color"] : "red";
        $mes = $_GET["message"];

        echo '<p style="color: '.$color.'">'.$mes.'</p>';
    }
}
```