Домашняя работа

по предмету: Основы PHP (семинары)

Выполнил: Байбородин Александр

Группа: Программист | 17.04.2024

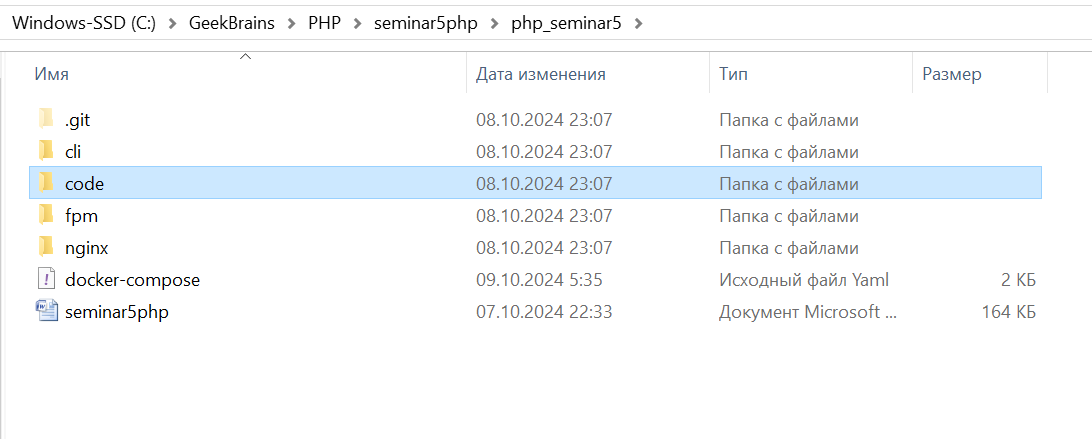
### Урок 5. Разработка каркаса приложения

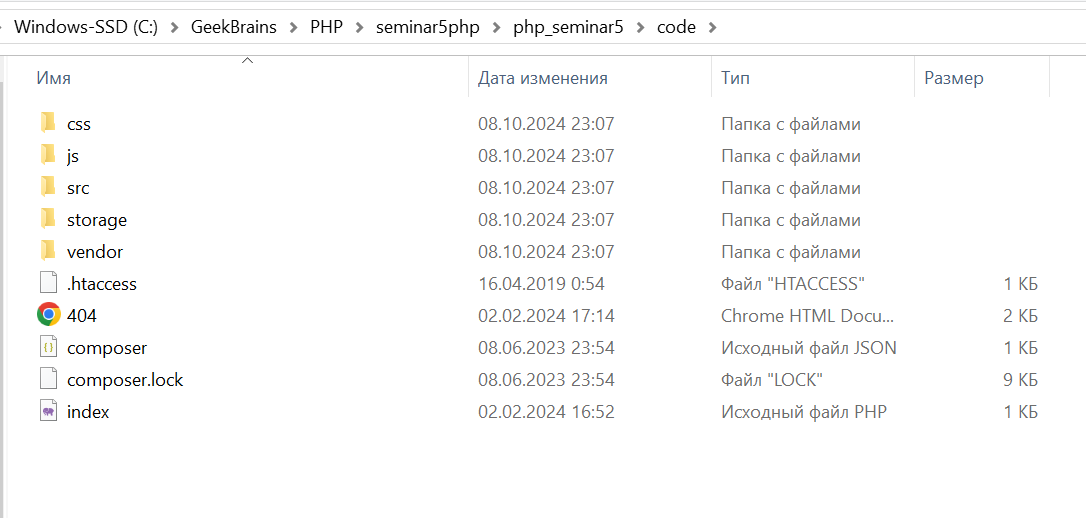
Задание

Добавьте к шаблону подключение файлов стилей так, чтобы в дальнейшем можно было дорабатывать внешний вид системы.  
  
Сформируйте еще три подключаемых к скелету блока – шапку сайта (она всегда будет одинаковой по стилю и располагаться в самой верхней части), подвал сайта (также одинаковый, но в нижней части) и sidebar (боковая колонка, которую можно наполнять новыми элементами).  
  
Средствами TWIG выводите на экран текущее время.  
  
Создайте обработку страницы ошибки. Например, если контроллер на найден, то нужно вызывать специальный метод рендеринга, который сформирует специальную страницу ошибок.  
  
Для страницы ошибок формируйте HTTP-ответ 404. Это можно сделать при помощи функции header.  
  
Реализуйте функционал сохранения пользователя в хранилище. Сохранение будет происходить при помощи GET-запроса.  
  
/user/save/?name=Иван&birthday=05-05-1991

Решение:

Сформируем структуру каталогов и файлов проекта:





Сформируем докер-файл, для сборки и запуска контейнера:

# версия синтаксиса

version: '3'

# в этом блоке мы описываем контейнеры, которые будут запускаться

services:

  #контейнер с Nginx

  nginx:

    build:

      context: ./nginx

      dockerfile: Dockerfile

    image: myapp/nginx

    container\_name: webserver

    # проброс портов

    ports:

      - "80:80"

    volumes:

      - ./code:/data/mysite.local

    networks:

      - app-network

  #Контейнер с PHP-FPM, назовём его app

  application:

    # Если нет секции build, то система будет искать образ в репозиториях

    build:

      context: ./fpm

      dockerfile: Dockerfile

    image: myapp/php # имя будущего образа

    container\_name: app # имя контейнера после запуска

    volumes:

      - ./code:/data/mysite.local

    # мы можем создать для контейнеров внутреннюю сеть

    networks:

      - app-network

  #Контейнер с PHP-FPM, назовём его app

  cli:

    # Если нет секции build, то система будет искать образ в репозиториях

    build:

      context: ./cli

      dockerfile: Dockerfile

    image: myapp/php-cli # имя будущего образа

    container\_name: cli # имя контейнера после запуска

    volumes:

      - ./code:/data/mysite.local

    # мы можем создать для контейнеров внутреннюю сеть

    networks:

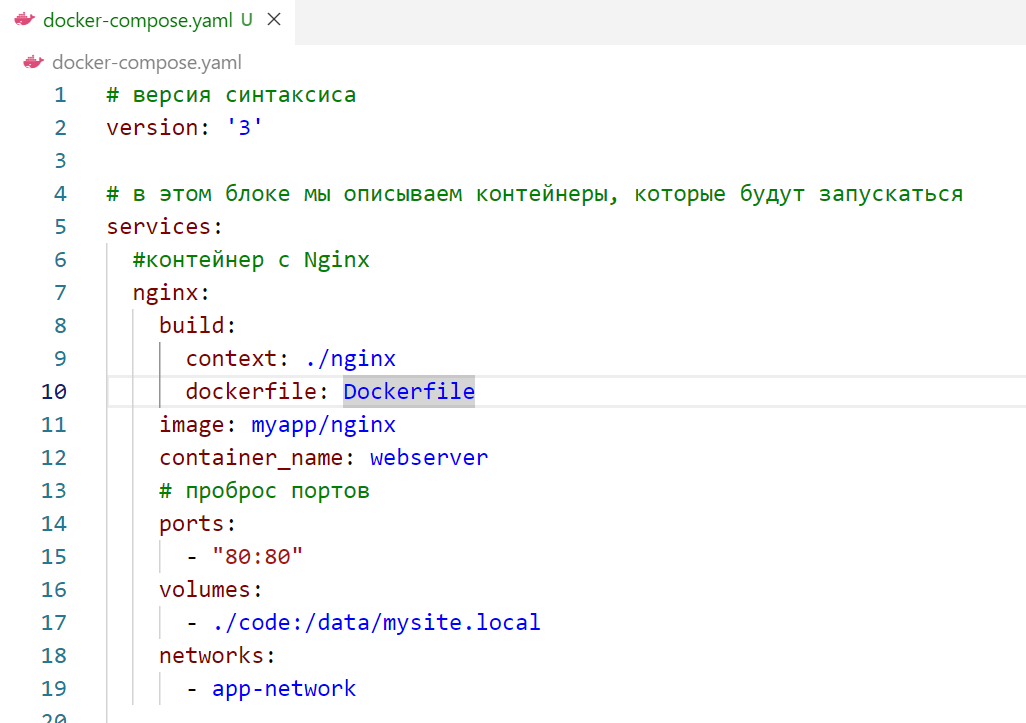
      - app-network

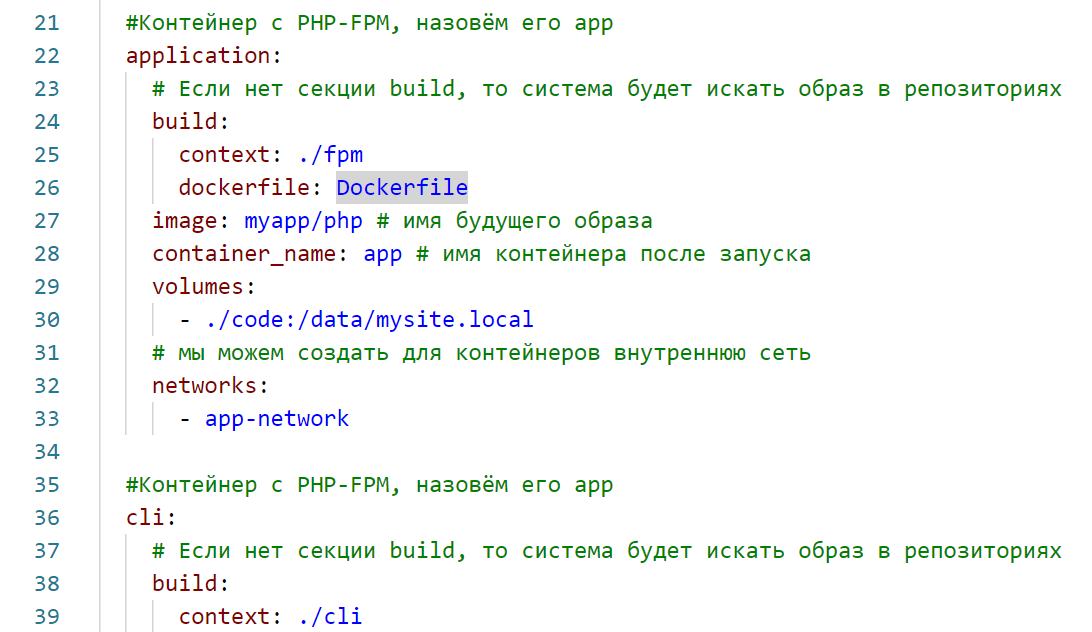
#Docker Networks

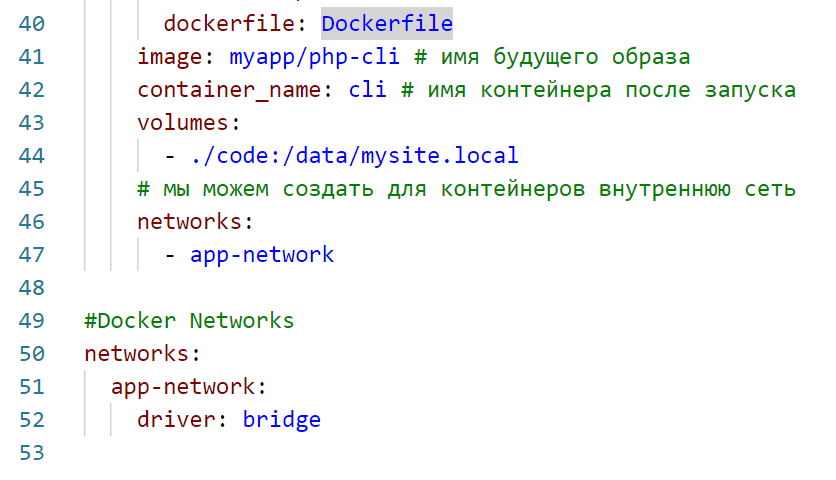
networks:

  app-network:

    driver: bridge







Добавим к шаблону подключение файлов стилей так, чтобы в дальнейшем можно было дорабатывать внешний вид системы, cформируем еще три подключаемых к скелету блока – шапку сайта (она всегда будет одинаковой по стилю и располагаться в самой верхней части), подвал сайта (также одинаковый, но в нижней части) и sidebar (боковая колонка, которую можно наполнять новыми элементами).:

<!DOCTYPE html>

<html>

<head>

<title>{{ title }}</title>

<link href="/css/main.css" rel="stylesheet">

</head>

<body>

{% include 'layouts/site-header.twig' %}

<nav>

<a href="/">Главная</a>

<a href="/user">Пользователи</a>

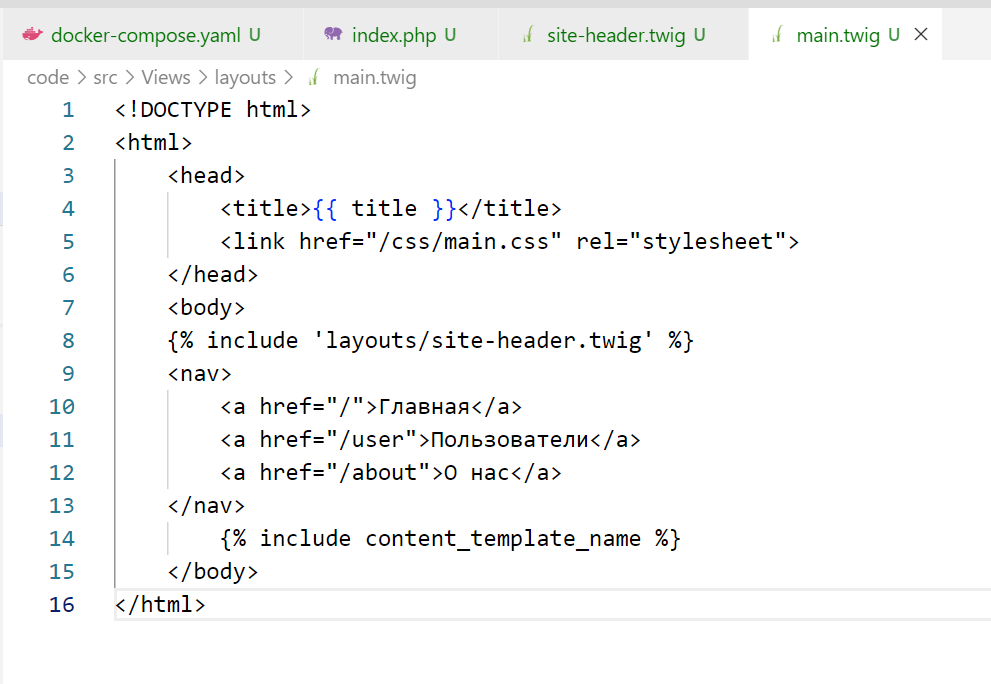
<a href="/about">О нас</a>

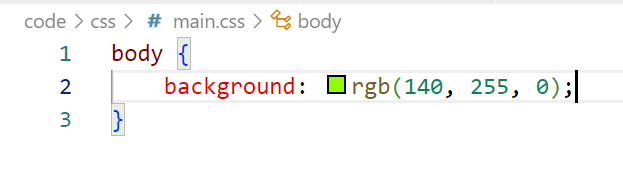
</nav>

{% include content\_template\_name %}

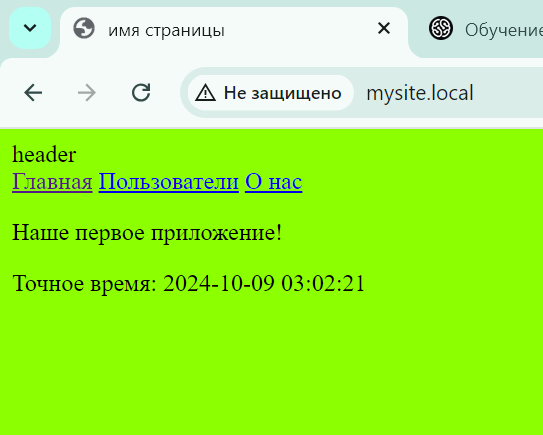
</body>

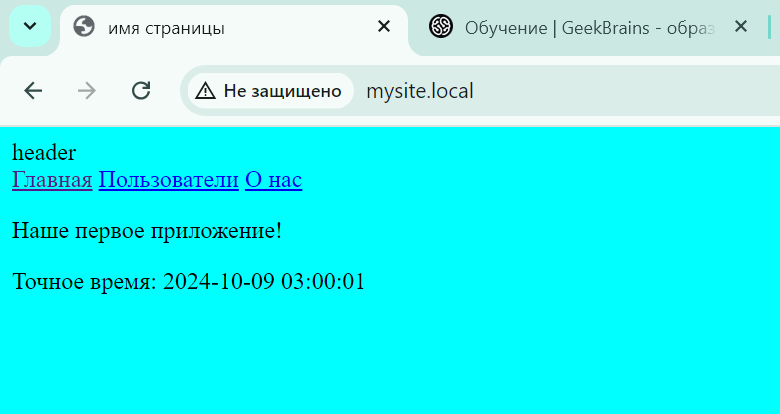
</html>



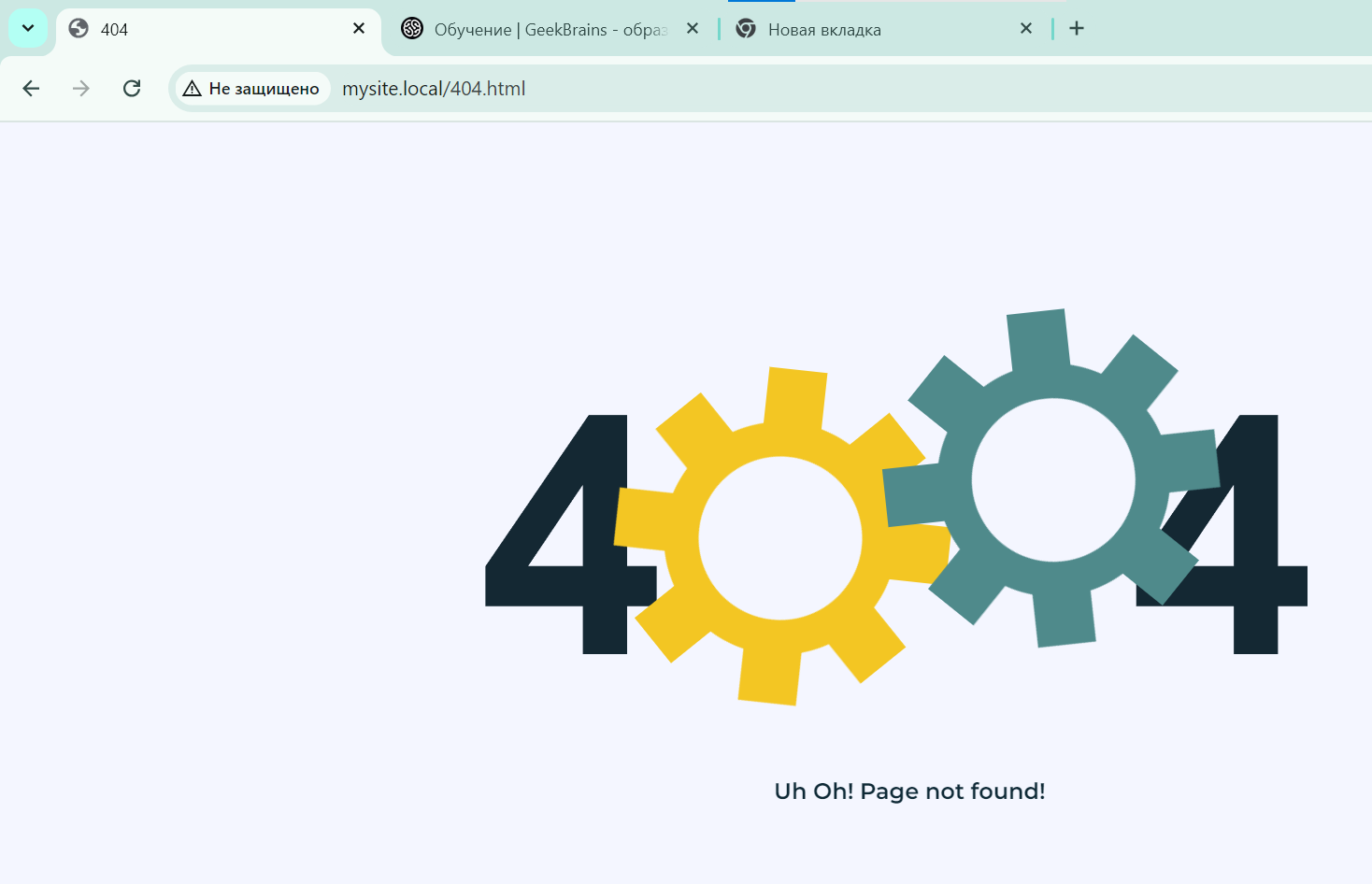


Средствами TWIG выведем на экран текущее время:





Создадим обработку страницы ошибки. Например, если контроллер на найден, то нужно вызывать специальный метод рендеринга, который сформирует специальную страницу ошибок. Для страницы ошибок сформируем HTTP-ответ 404 :



<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>404</title>

    <link rel="stylesheet" href="css/404.css">

    <script src="https://cdnjs.cloudflare.com/ajax/libs/gsap/3.12.5/gsap.min.js"></script>

</head>

<body>

<div class="container">

    <h1 class="first-four">4</h1>

    <div class="cog-wheel1">

        <div class="cog1">

            <div class="top"></div>

            <div class="down"></div>

            <div class="left-top"></div>

            <div class="left-down"></div>

            <div class="right-top"></div>

            <div class="right-down"></div>

            <div class="left"></div>

            <div class="right"></div>

        </div>

    </div>

    <div class="cog-wheel2">

        <div class="cog2">

            <div class="top"></div>

            <div class="down"></div>

            <div class="left-top"></div>

            <div class="left-down"></div>

            <div class="right-top"></div>

            <div class="right-down"></div>

            <div class="left"></div>

            <div class="right"></div>

        </div>

    </div>

    <h1 class="second-four">4</h1>

    <p class="wrong-para">Uh Oh! Page not found!</p>

    <script src="js/404.js"></script>

</div>

</body>

</html>

@import url("https://fonts.googleapis.com/css2?family=Poppins:wght@700&display=swap" rel="stylesheet");

@import url("https://fonts.googleapis.com/css?family=Montserrat:400,500,600,700");

\*{

    margin:0;

    padding:0;

    box-sizing:border-box;

}

body{

    overflow:hidden;

    background-color: #f4f6ff;

}

.container{

    width:100vw;

    height:100vh;

    display: flex;

    justify-content: center;

    align-items: center;

    font-family: "Poppins", sans-serif;

    position: relative;

    left:6vmin;

    text-align: center;

}

.cog-wheel1, .cog-wheel2{

    transform:scale(0.7);

}

.cog1, .cog2{

    width:40vmin;

    height:40vmin;

    border-radius:50%;

    border:6vmin solid #f3c623;

    position: relative;

}

.cog2{

    border:6vmin solid #4f8a8b;

}

.top, .down, .left, .right, .left-top, .left-down, .right-top, .right-down{

    width:10vmin;

    height:10vmin;

    background-color: #f3c623;

    position: absolute;

}

.cog2 .top,.cog2  .down,.cog2  .left,.cog2  .right,.cog2  .left-top,.cog2  .left-down,.cog2  .right-top,.cog2  .right-down{

    background-color: #4f8a8b;

}

.top{

    top:-14vmin;

    left:9vmin;

}

.down{

    bottom:-14vmin;

    left:9vmin;

}

.left{

    left:-14vmin;

    top:9vmin;

}

.right{

    right:-14vmin;

    top:9vmin;

}

.left-top{

    transform:rotateZ(-45deg);

    left:-8vmin;

    top:-8vmin;

}

.left-down{

    transform:rotateZ(45deg);

    left:-8vmin;

    top:25vmin;

}

.right-top{

    transform:rotateZ(45deg);

    right:-8vmin;

    top:-8vmin;

}

.right-down{

    transform:rotateZ(-45deg);

    right:-8vmin;

    top:25vmin;

}

.cog2{

    position: relative;

    left:-10.2vmin;

    bottom:10vmin;

}

h1{

    color:#142833;

}

.first-four{

    position: relative;

    left:6vmin;

    font-size:40vmin;

}

.second-four{

    position: relative;

    right:18vmin;

    z-index: -1;

    font-size:40vmin;

}

.wrong-para{

    font-family: "Montserrat", sans-serif;

    position: absolute;

    bottom:15vmin;

    padding:3vmin 12vmin 3vmin 3vmin;

    font-weight:600;

    color:#092532;

}

Реализуем функционал сохранения пользователя в хранилище. Сохранение будет происходить при помощи GET-запроса.

Проверяем, что при GET-запросе происходит сохранение пользователя в хранилище:

