

**Санкт-Петербургский Национальный Исследовательский
Университет Информационных технологий, механики и оптики**

Отчет

Дисциплина: Web-программирование

Лабораторная работа №3

Выполнил: Смирнов И.И.

Группа № K3321

Проверил:

Марченко Е.В.

Санкт-Петербург

2024

Оглавление

Оглавление.....	2
Введение.....	3
Ход работы.....	3
Задание 1	3
Задание 2	5
Задание 3	7
Вывод	11

Введение

Целью данной лабораторной работы является изучение инструментов web-разработки

Были выполнены следующие задачи:

- Сделать последовательное и параллельное выполнение task в gulp. Также создать task, который автоматически обновляет страницу сайта во время разработки;
- Создать рабочую форму обратной связи, работающую с get и post запросами;
- Запустить движок с Wordpress и запустить на портал <http://test.site>.

Ход работы

Задание 1

Используя знания gulp из лабораторной работы №2 был создан проект и загружен в него gulp. Для пункта а) в gulpfile.js были созданы 2 простые задачи, которые выводят в консоль строку (рисунок 1).

```
const {series, parallel} = require('gulp');

function hello(cb) {
  console.log("hello!");
  cb();
}

function bye(cb) {
  console.log('bye bye');
  cb();
}

exports.ser = series(hello, bye);
exports.par = parallel(hello, bye);
```

Рисунок 1 – Gulpfile.js пункт а

Далее были вызваны задачи ser, последовательная работа задач, и par, параллельная работа задач (рисунок 2).

```

C:\Users\igors\VSCodeProjects\webLabs\lab3>gulp ser
[19:23:14] Using gulpfile ~\VSCodeProjects\webLabs\lab3\gulpfile.js
[19:23:14] Starting 'ser'...
[19:23:14] Starting 'hello'...
hello!
[19:23:14] Finished 'hello' after 2.22 ms
[19:23:14] Starting 'bye'...
bye bye
[19:23:14] Finished 'bye' after 1.66 ms
[19:23:14] Finished 'ser' after 9.48 ms

C:\Users\igors\VSCodeProjects\webLabs\lab3>gulp par
[19:23:25] Using gulpfile ~\VSCodeProjects\webLabs\lab3\gulpfile.js
[19:23:25] Starting 'par'...
[19:23:25] Starting 'hello'...
[19:23:25] Starting 'bye'...
hello!
[19:23:25] Finished 'hello' after 2.23 ms
bye bye
[19:23:25] Finished 'bye' after 2.32 ms
[19:23:25] Finished 'par' after 6.85 ms

```

Рисунок 2 – последовательное и параллельное выполнение

Для реализации пункта б) был подключен Browsersync. На основе его функций была написана третья задача в gulpfile.js (Рисунок 3).

```

1  const gulp = require('gulp');
2
3  const browserSync = require('browser-sync').create();
4
5  function serve() {
6    browserSync.init({
7      server: {
8        baseDir: 'C:/Users/igors/VSCodeProjects/webLabs/lab3/task1'
9      }
10   });
11
12   gulp.watch('*.html').on('change', browserSync.reload);
13   gulp.watch('css/*.css').on('change', browserSync.reload);
14   gulp.watch('js/*.js').on('change', browserSync.reload);
15 }
16
17
18 exports.serve = serve;
19 exports.default = serve;
20
21

```

Рисунок 3 – Gulpfile.js пункт б)

Для проверки создан html-документ, содержащий в себе 1 h1 заголовок. После этого был запущен gulp task и внесено изменение в виде добавления h2 заголовка в документ. Задача заметила это изменение и обновила страницу в браузере сразу после сохранения документа (Рисунок 4).

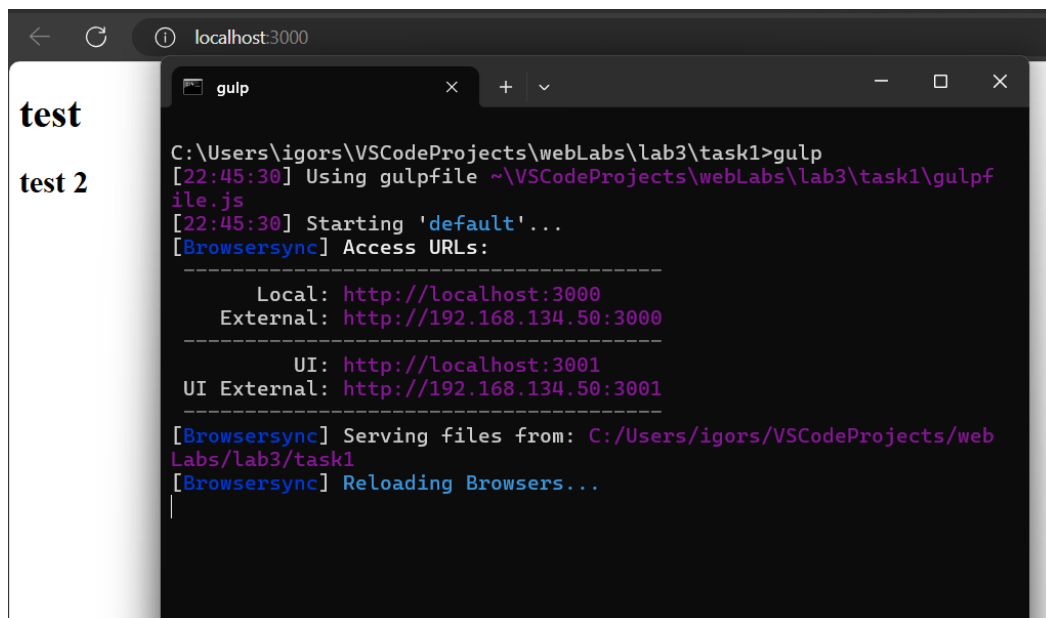


Рисунок 4 – Обновление страницы в браузере

Задание 2

В этом задании необходимо создать форму обратной связи и отправить полученные от пользователя данные через POST и GET запрос. Для этого была написана простая html страница без применения css (рисунок 5) и скрипт на php (рисунок 6).

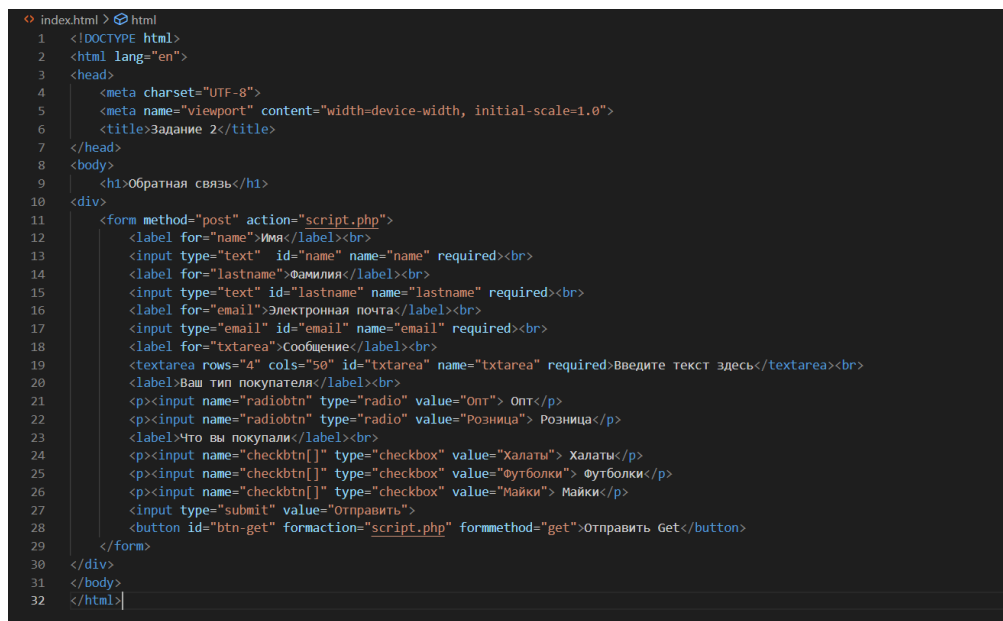


Рисунок 5 – Разметка страницы обратной связи

```

script.php > ...
1  <?php
2  if ($ _SERVER['REQUEST_METHOD'] == 'POST') {
3      $name = htmlspecialchars(string: $_POST['name']);
4      $last_name = htmlspecialchars(string: $_POST['lastname']);
5      $email = htmlspecialchars(string: $_POST['email']);
6      $msg = htmlspecialchars(string: $_POST['txtarea']);
7      $radiobtn = htmlspecialchars(string: $_POST['radiobtn']);
8      $checkboxbtn = isset($_POST['checkboxbtn']) ? $_POST['checkboxbtn'] : [];
9      echo "<h1>Спасибо! Форма отправлена</h1>";
10     echo "<p>Имя: $name </p>";
11     echo "<p>Фамилия: $last_name </p>";
12     echo "<p>Электронная почта: $email </p>";
13     echo "<p>Отзыв: $msg </p>";
14     echo "<p>Размер покупки: $radiobtn </p>";
15     if (!empty($checkboxbtn)) {
16         echo "<p> Купленный товар: " . implode(separator: ", ", array: $checkboxbtn) . "</p>";
17         echo "<p> Купленный товар: не указано";}
18     } elseif ($ _SERVER["REQUEST_METHOD"] == "GET") {
19         $name = htmlspecialchars(string: $_GET['name']);
20         $last_name = htmlspecialchars(string: $_GET['lastname']);
21         $email = htmlspecialchars(string: $_GET['email']);
22         $msg = htmlspecialchars(string: $_GET['txtarea']);
23         $radiobtn = htmlspecialchars(string: $_GET['radiobtn']);
24         $checkboxbtn = isset($_GET['checkboxbtn']) ? $_GET['checkboxbtn'] : [];
25         echo "<h1>Спасибо! Форма отправлена</h1>";
26         echo "<p>Имя: $name </p>";
27         echo "<p>Фамилия: $last_name </p>";
28         echo "<p>Электронная почта: $email </p>";
29         echo "<p>Отзыв: $msg </p>";
30         echo "<p>Размер покупки: $radiobtn </p>";
31         if (!empty($checkboxbtn)) {
32             echo "<p> Купленный товар: " . implode(separator: ", ", array: $checkboxbtn) . "</p>";
33             echo "<p> Купленный товар: не указано";}
34     } else {
35         echo "<h1> Ошибка </h1>";
36     }
37 }

```

Рисунок 6 – PHP-скрипт

Далее была протестирована работа формы. После нажатия на обе кнопки, отображается информация, согласно инструкциям в скрипте (рисунок 7).

Спасибо! Форма отправлена

Имя: Игорь

Фамилия: Смирнов

Электронная почта: igorsmirnov604@gmail.com

Отзыв: супер топ

Размер покупки: Опт

Купленный товар: Футболки, Майки

Рисунок 7 – Результат работы скрипта

Визуально get и post запросы не отличаются. Но отображения адреса страницы имеют различия: Get запрос все полученные аргументы отображает в URL (рисунок 8), а post запрос скрывает эту информацию (рисунок 9).

localhost/taskphp/script.php?name=Игорь&lastname=Смирнов&email=igorsmirnov604%40gmail.com&txtarea=сунер+тон&radiobtn=Отт&checkboxbtn%5B%5D=Футболки&che...

Рисунок 8 – URL при get запросе

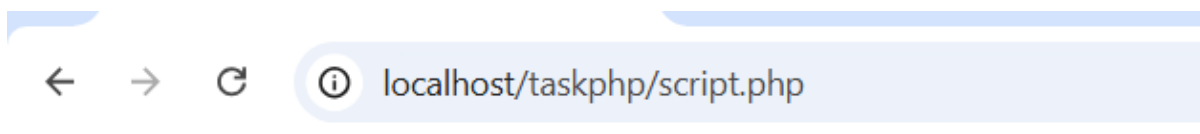


Рисунок 9 –URL при post запросе

Задание 3

В задании необходимо установить локальный сервер на свое усмотрение. Во втором задании для работы php был установлен wamp. Он же и будет взят в этом задании. Также был скачен Wordpress и директория с ним (task3) помещена в директорию wamp (wamp64/www). В адресной строке был прописан адрес localhost/task3 и началась первоначальная настройка. Под сайт была необходима база данных, она была создана по с помощью адреса <http://localhost/phpmyadmin> (рисунок10).

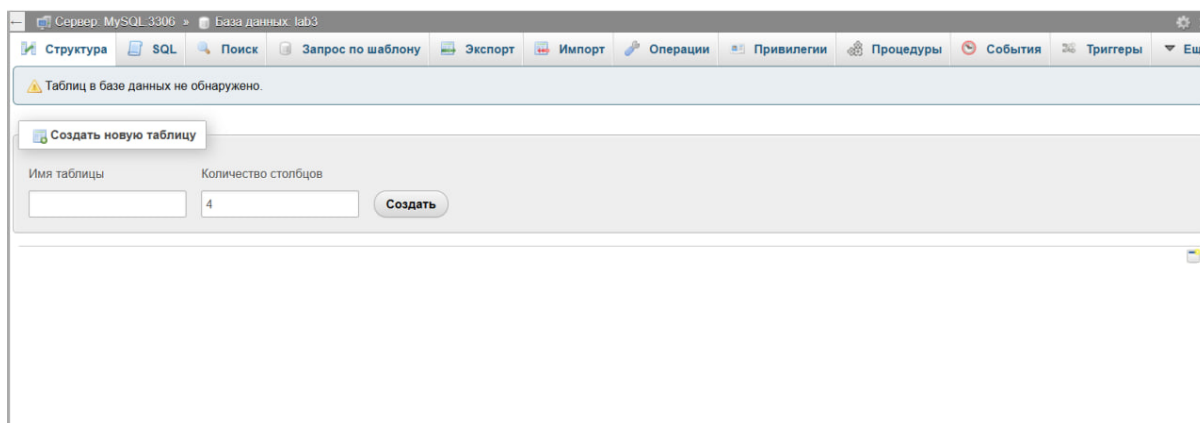


Рисунок 10 – создание БД lab3

Далее была заполнена информация по сайту (рисунок 11).

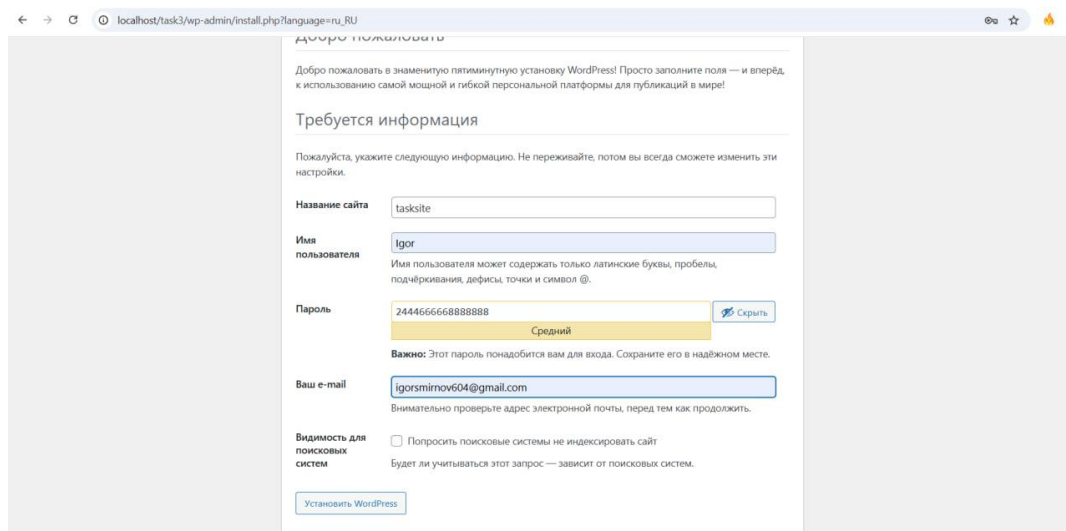


Рисунок 11 – создание сайта

После успешной установки в браузере открылась главная страница админки wordpress (рисунок 12).

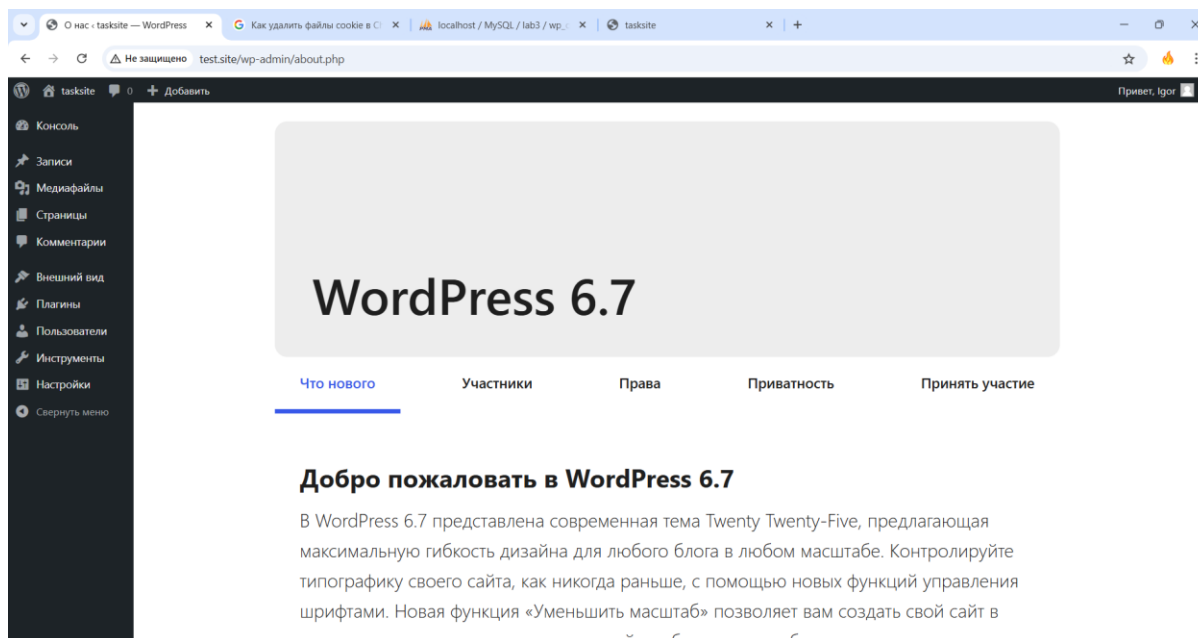
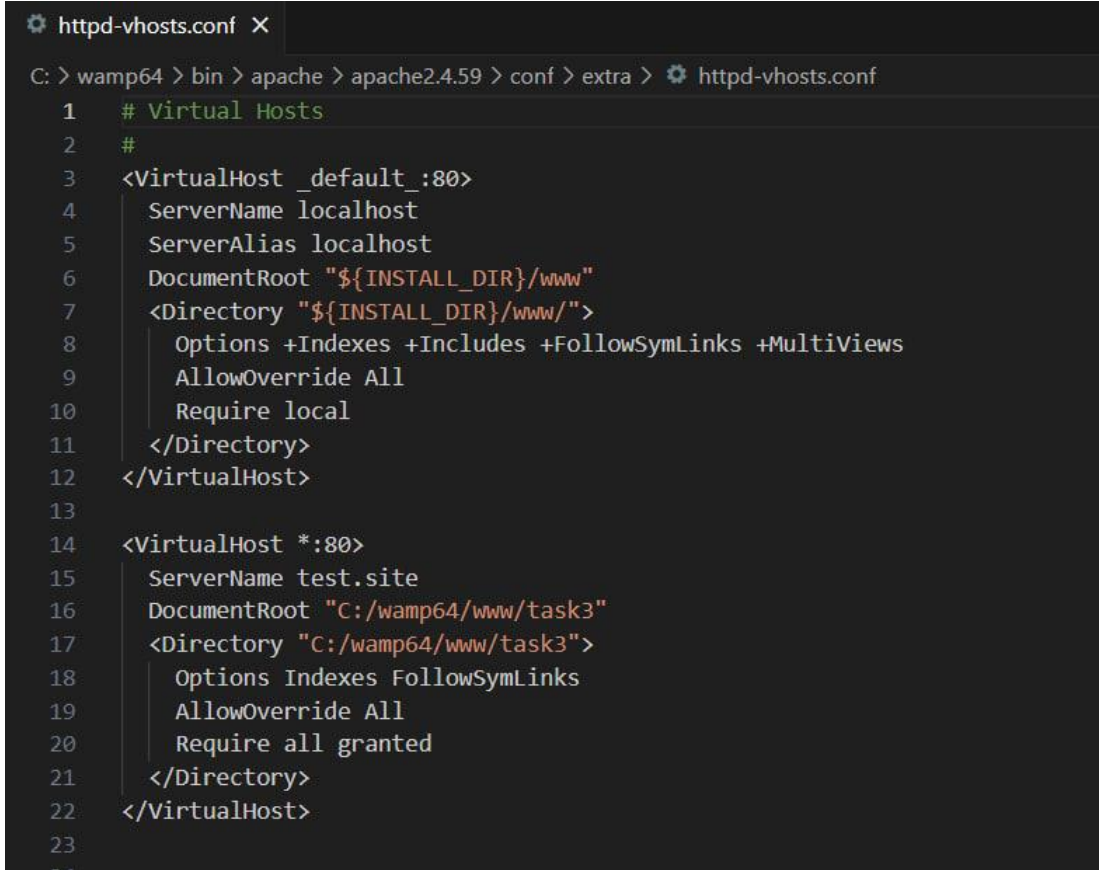


Рисунок 12 – страница wordpress

В задании необходимо открыть портал <http://test.site>. Для этого необходимо произвести несколько манипуляций в файлах проекта:

- 1) была добавлена строчка “127.0.0.1 test.site” в файл hosts, расположенный по адресу C:\Windows\System32\drivers\etc;
- 2) В файле httpd.conf по адресу C:\wamp64\bin\apache\apache2.4.59\conf было разрешено использование виртуальных хостов;

3) В файле httpd-vhosts.conf по адресу
C:\wamp64\bin\apache\apache2.4.59\conf\extra был настроен локальный хост
– строки 14-22 (рисунок 13).



```
1  # Virtual Hosts
2  #
3  <VirtualHost _default_:80>
4      ServerName localhost
5      ServerAlias localhost
6      DocumentRoot "${INSTALL_DIR}/www"
7      <Directory "${INSTALL_DIR}/www/">
8          Options +Indexes +Includes +FollowSymLinks +MultiViews
9          AllowOverride All
10         Require local
11     </Directory>
12 </VirtualHost>
13
14 <VirtualHost *:80>
15     ServerName test.site
16     DocumentRoot "C:/wamp64/www/task3"
17     <Directory "C:/wamp64/www/task3">
18         Options Indexes FollowSymLinks
19         AllowOverride All
20         Require all granted
21     </Directory>
22 </VirtualHost>
23
24
```

Рисунок 13 – настройка виртуального хоста

После этих действий был перезагружен wamp и в phpMyAdmin в базе данных lab3 в таблице wp_options значение полей siteurl и home было изменено на test.site (рисунок 14).

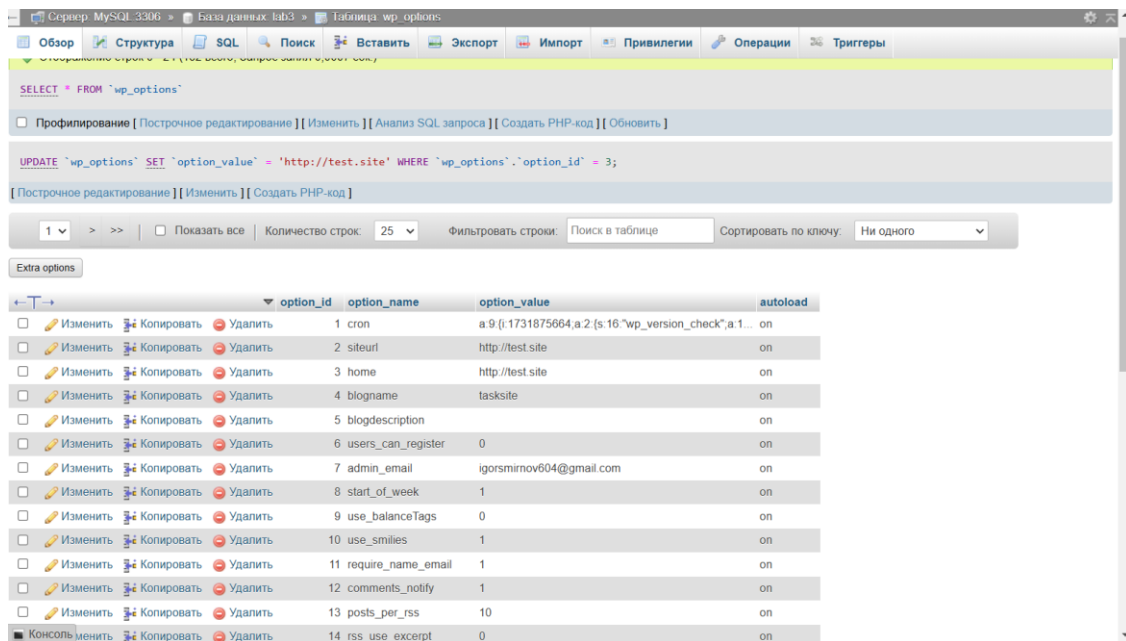


Рисунок 14 – Изменение адреса в БД

После всех действий портал <http://test.site> успешно открылся в браузере (рисунок 15).

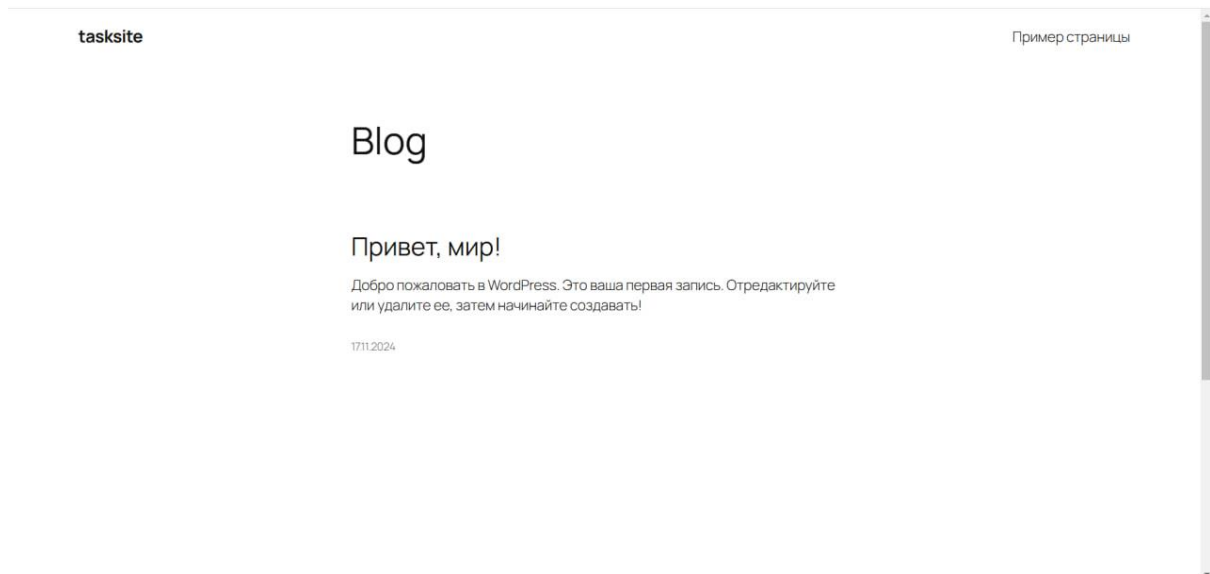


Рисунок 15 – страница в браузере

В wordpress в соответствующем разделе была выбрана новая тема для данной страницы и после перезагрузки страницы тема применилась. Новая тема: Automobile Elementor (рисунок 16).

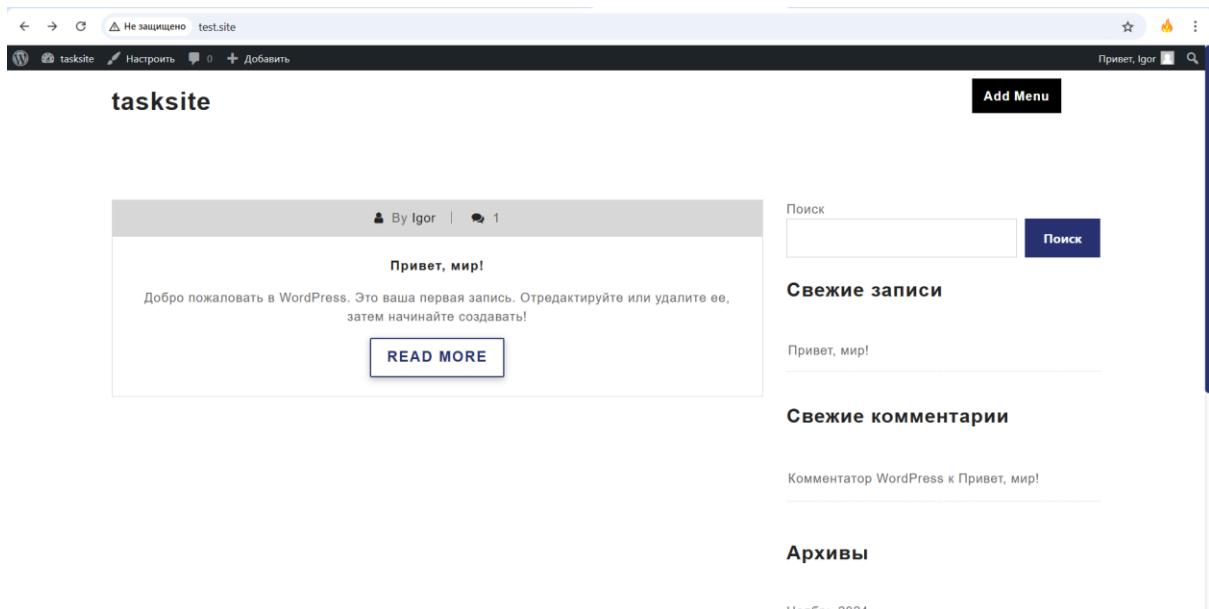


Рисунок 16 – применение пользовательской темы

Вывод

В данной лабораторной работе были дополнены знания по использованию gulp, а также произведено знакомство с get и post запросами, языком php, средствами wamp и wordpress.