

**Министерство науки и высшего образования Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
ITMO University**

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ 3

По дисциплине Web-программирование

Тема работы Создание сайта по отправлению обратной связи

Обучающийся Алексеев Тимофей Юрьевич

Факультет Факультет инфокоммуникационных технологий

Группа K3221

Направление подготовки 11.03.02 Инфокоммуникационные технологии и системы связи

Образовательная программа Программирование в инфокоммуникационных системах

Обучающийся	<u>11.11.2024</u> (дата)	<u> </u> (подпись)	<u>Алексеев Т.Ю.</u> (Ф.И.О.)
Руководитель	<u> </u> (дата)	<u> </u> (подпись)	<u>Марченко Е.В.</u> (Ф.И.О.)

Санкт-Петербург
2024 г.

Цель

Создать сайт для отправки информации по обратной связи, используя для этого gulp, php-скрипт и инструментальный для отладки проектов.

Задачи

1. Ознакомиться с параллельными и последовательными задачами Gulp;
2. Написать форму обратной связи с помощью PHP-скриптов;
3. Установить WordPress и настроить переход на сайт по ссылке test.site.

Ход работы

Задание 1

В данном задании было необходимо произвести работу с последовательными и параллельными функциями.

Для начала было необходимо создать одну параллельную и одну последовательную задачу. В Gulp за это отвечают `series()` и `parallel()`.

Для выполнения данной задачи были написаны две задачи: `hello` и `bye`, которые выводят приветственное и прощальное сообщения соответственно. После чего были созданы задачи, включающие две вышеупомянутые задачи и использующие `series()` и `parallel()`, - `orderTasks()` и `parallelTasks()`.

```
var gulp = require('gulp');
const browserSync = require('browser-sync').create();

gulp.task('hello', function(done) {
  console.log('Hello, my dear editor!');
  done();
});

gulp.task('bye', function(done) {
  console.log('Bye Bye, my dear editor!');
  done();
});

gulp.task('orderTasks', gulp.series('hello', 'bye'));
gulp.task('parallelTasks', gulp.parallel('hello', 'bye'));
```

Рисунок 1 – Параллельные и последовательные задачи

Проверим, что задачи выполняются ровно в том порядке, как мы и хотим. Для этого вызовем их в консоли с помощью `npm run orderTasks` и `npm run parallelTasks`.

```

_tasks>npm run orderTasks

> gulp_tasks@1.0.0 orderTasks
> gulp orderTasks

[23:52:48] Using gulpfile ~\OneDrive\Рабочий стол\lab_3\gulp_tasks\gulpfile.js
[23:52:48] Starting 'orderTasks'...
[23:52:48] Starting 'hello'...
Hello, my dear editor!
[23:52:48] Finished 'hello' after 1.56 ms
[23:52:48] Starting 'bye'...
Bye Bye, my dear editor!
[23:52:48] Finished 'bye' after 1.18 ms
[23:52:48] Finished 'orderTasks' after 7.91 ms

```

Рисунок 2 – Последовательное выполнение задач

Как мы видим, задачи действительно выполняются по очереди, когда заканчивает выполнение одна, вторая сразу же начинает своё выполнение.

```

_tasks>npm run parallelTasks

> gulp_tasks@1.0.0 parallelTasks
> gulp parallelTasks

[23:54:49] Using gulpfile ~\OneDrive\Рабочий стол\lab_3\gulp_tasks\gulpfile.js
[23:54:49] Starting 'parallelTasks'...
[23:54:49] Starting 'hello'...
[23:54:49] Starting 'bye'...
Hello, my dear editor!
[23:54:49] Finished 'hello' after 2.84 ms
Bye Bye, my dear editor!
[23:54:49] Finished 'bye' after 3.28 ms
[23:54:49] Finished 'parallelTasks' after 8.11 ms

```

Рисунок 3 – Параллельное выполнение задач

Здесь же ситуация обратная. Сначала запускаются все задачи, после чего происходит их одновременное выполнение.

После этого необходимо было создать задачу, при выполнении которой в браузере отображались бы файлы проекта и происходило обновление при каждом изменении.

Для этого в папке app были созданы файлы index.html с тегом <h2> и текстом Hello world и script.js с выводом аналогичного текста в консоль.

Далее были созданы задачи html, script и server.

В задачах `html` и `script` просто добавляются соответствующие файлы в поток для отображения их в браузере с помощью `browserSync`.

В задаче `server` запускается локальный сервер и папка, в которой он базируется. А также прописываются методы `watch`, следящие за изменениями в `html`- и `js`-файлах и перезагружающие страницу, если таковые произошли.

После чего была написана главная задача `startWeb`, объединяющая все подзадачи. В ней последовательно вызываются параллельное выполнение задач `script` и `html`, а потом выполнение задачи `server`.

```
gulp.task('html', function () {
  return gulp.src('app/index.html').pipe(browserSync.stream());
});

gulp.task('scripts', function () {
  return gulp.src('app/script.js').pipe(browserSync.stream());
});

gulp.task('server', function () {
  browserSync.init({
    server: {
      baseDir: 'app'
    }
  });

  gulp.watch('app/index.html').on('change', browserSync.reload);
  gulp.watch('app/script.js').on('change', browserSync.reload);
});

gulp.task('startWeb', gulp.series(gulp.parallel('html', 'scripts'), 'server'));
```

Рисунок 4 – Gulpfile для отображения файлов проекта

При выполнении задачи `startWeb` будет открываться следующая страница. При внесении изменений в файлы страница автоматически обновляется.

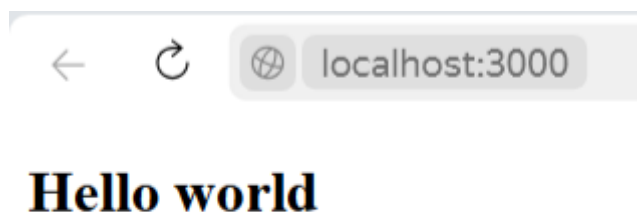


Рисунок 5 – Открывающаяся страница в браузере

Задание 2

В этом задании было необходимо реализовать форму обратной связи, которую можно было бы отправить и получить информацию с помощью php-скрипта.

Для начала необходимо было запустить локальный сервер. Для этой задачи я выбрал XAMPP из-за простоты работы с ним, графического интерфейса и прозрачной настройки. Для запуска проекта необходимо перенести все файлы запускаемого проекта в папку htdocs.

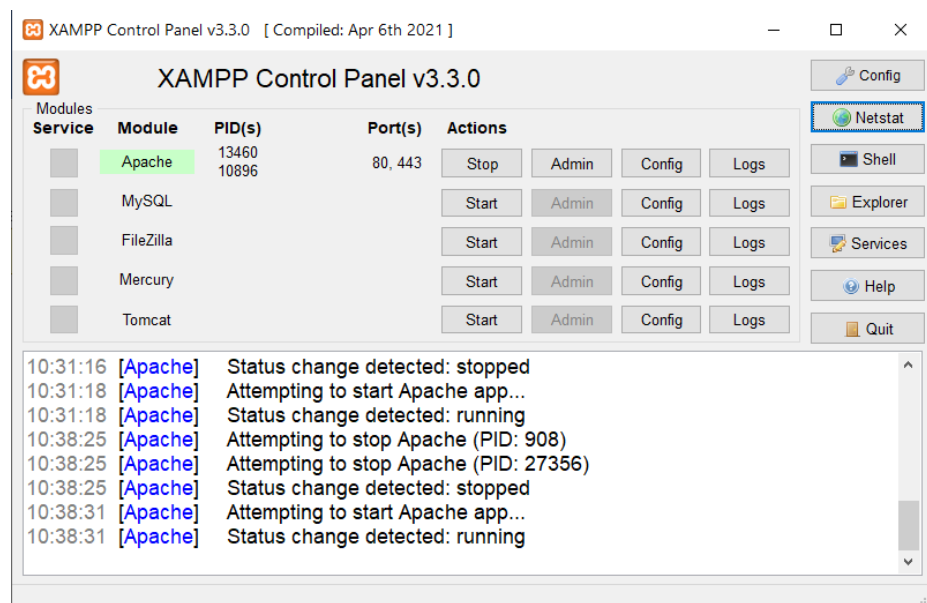


Рисунок 5 – Запуска локального сервера с помощью XAMPP

После этого был написан index.html, который и содержал начальную форму. Там пользователю предлагается заполнить имя, фамилию, почту, развернутую обратную связь в textarea, выбрать общее впечатление от мероприятия с помощью radio-input и выбрать понравившиеся аспекты с помощью checkbox-input. Форма отправляется по нажатию на соответствующую кнопку.

```

<!DOCTYPE html>
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Форма</title>
</head>
<body>
  <h2>Форма обратной связи</h2>
  <form action="start_post.php" method="post">
    <p>Имя:</p>
    <input type="text" name="first-name">
    <p>Фамилия:</p>
    <input type="text" name="second-name">
    <p>Email:</p>
    <input type="email" name="email">
    <p>Обратная связь:</p>
    <textarea name="feedback" cols="20" row="20"></textarea>

    <p>Оцените общее впечатление от мероприятия от 0 до 5:</p>
    <p><input type="radio" name="experience" value="0">0</p>
    <p><input type="radio" name="experience" value="1">1</p>
    <p><input type="radio" name="experience" value="2">2</p>
    <p><input type="radio" name="experience" value="3">3</p>
    <p><input type="radio" name="experience" value="4">4</p>
    <p><input type="radio" name="experience" value="5">5</p>

    <p>Выберите аспекты мероприятия, которые понравились вам особенно:</p>
    <input type="checkbox" name="point[]" value="competition" />Конкурсная программа<br>
    <input type="checkbox" name="point[]" value="meal" />Фуршет<br>
    <input type="checkbox" name="point[]" value="education" />Образовательная часть<br>
    <input type="checkbox" name="point[]" value="party" />Вечеринка после закрытия<br>
    <br>
    <input type="submit" value="Отправить">
  </form>
</body>

```

Рисунок 6 – HTML-код формы обратной связи

Далее необходимо написать PHP-скрипт, который обрабатывает post- и get-запросы. В post-запросе сначала создаются все переменные, которые были получены после отправления формы. Поле со множественным выбором сначала проверяется на пустоту, после чего идет подсчет выбранных аспектов. После этого происходит запись в txt-файл.

В get-запросе обратная логика. Он вызывается при нажатии на кнопку на новой отрисованной странице с заголовком «Спасибо за ваш отзыв!». После этого считывается вся информация из txt-файла и выводится на экран пользователя. Выводится это с помощью тега <pre> для сохранения исходного форматирования.

```

<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $name = $_POST['first-name'];
    $surname = $_POST['second-name'];
    $email = $_POST['email'];
    $feedback = $_POST['feedback'];
    $experience = $_POST['experience'];
    $points = count(isset($_POST['point']) ? $_POST['point'] : []);

    $data = "Имя: $name\nФамилия: $surname\nEmail: $email\nОбратная связь: $feedback\nОбщая оценка: $experience";

    file_put_contents('data.txt', $data, FILE_APPEND);
}
?>

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Обработанная обратная связь</title>
    <style> pre { font-family: 'Times New Roman', serif; font-size: 16px; } </style>
</head>
<body>
    <?php
    if ($_SERVER["REQUEST_METHOD"] == "GET") {
        $fileContent = file_get_contents('data.txt');
        echo '<h1>Ваш отзыв:</h1>';
        echo "<pre>$fileContent</pre>";
    } else {
        echo '<h1>Спасибо за ваш отзыв!</h1>';
        echo '<br>';
        echo '<form action="start_post.php" method="get">';
        echo '<input type="submit" value="Посмотреть ответы">';
        echo '</form>';
    }
    ?>
</body>
</html>

```

Рисунок 7 – Код PHP-скрипта

В данный момент стартовая страница выглядит следующим образом.

← ↻ localhost

Форма обратной связи

Имя:

Фамилия:

Email:

Обратная связь:

Оцените общее впечатление от мероприятия от 0 до 5:

☐ 0
☐ 1
☐ 2
☐ 3
☐ 4
☐ 5

Выберите аспекты мероприятия, которые понравились вам особенно:

☐ Конкурсная программа
☐ Фуршет
☐ Образовательная часть
☐ Вечеринка после закрытия

Рисунок 8 – Страница сбора обратной связи

После отправки информации появляется следующая страница.

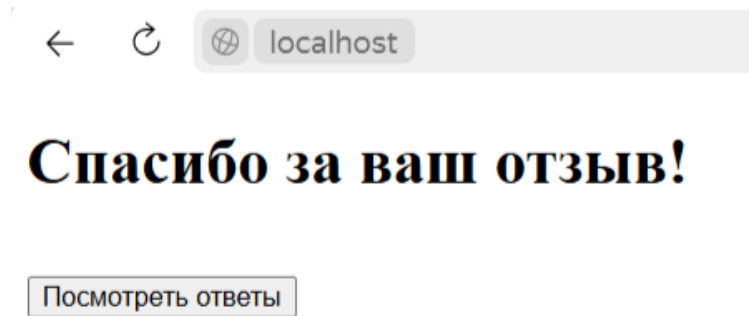


Рисунок 9 – Страница-ответ на отправку формы

И при нажатии на кнопку «Посмотреть ответы» открывается страница с формой пользователя.

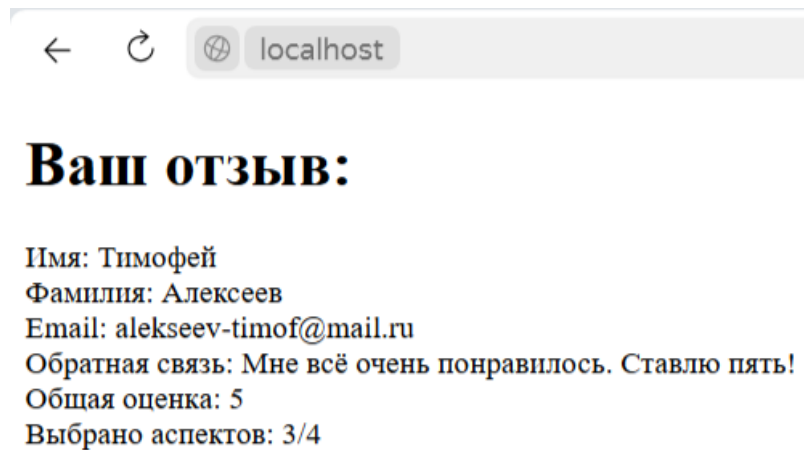


Рисунок 10 – Страница с пользовательским запросом

Задание 3

Для выполнения данного задания необходимо было использовать ранее установленный ХМАРР. Было произведено скачивание движка wordpress для его дальнейшего запуска. Папка была перемещена в папку htdocs для дальнейшего запуска.

После этого необходимо было перейти по ссылке <http://localhost/phpmyadmin>. Там происходило создание базы данных weblab3 для нашего сайта.

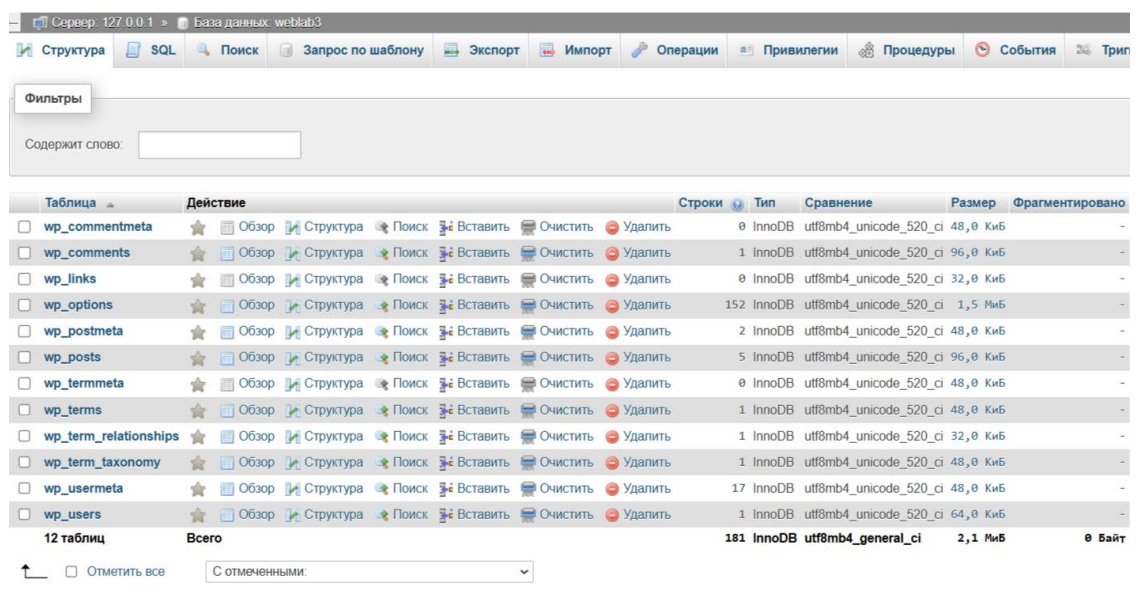


Таблица	Действие	Строки	Тип	Сравнение	Размер	Фрагментировано
<input type="checkbox"/> wp_commentmeta	Обзор Структура Поиск Вставить Очистить Удалить	0	InnoDB	utf8mb4_unicode_520_ci	48,0 Киб	-
<input type="checkbox"/> wp_comments	Обзор Структура Поиск Вставить Очистить Удалить	1	InnoDB	utf8mb4_unicode_520_ci	96,0 Киб	-
<input type="checkbox"/> wp_links	Обзор Структура Поиск Вставить Очистить Удалить	0	InnoDB	utf8mb4_unicode_520_ci	32,0 Киб	-
<input type="checkbox"/> wp_options	Обзор Структура Поиск Вставить Очистить Удалить	152	InnoDB	utf8mb4_unicode_520_ci	1,5 Миб	-
<input type="checkbox"/> wp_postmeta	Обзор Структура Поиск Вставить Очистить Удалить	2	InnoDB	utf8mb4_unicode_520_ci	48,0 Киб	-
<input type="checkbox"/> wp_posts	Обзор Структура Поиск Вставить Очистить Удалить	5	InnoDB	utf8mb4_unicode_520_ci	96,0 Киб	-
<input type="checkbox"/> wp_termmeta	Обзор Структура Поиск Вставить Очистить Удалить	0	InnoDB	utf8mb4_unicode_520_ci	48,0 Киб	-
<input type="checkbox"/> wp_terms	Обзор Структура Поиск Вставить Очистить Удалить	1	InnoDB	utf8mb4_unicode_520_ci	48,0 Киб	-
<input type="checkbox"/> wp_term_relationships	Обзор Структура Поиск Вставить Очистить Удалить	1	InnoDB	utf8mb4_unicode_520_ci	32,0 Киб	-
<input type="checkbox"/> wp_term_taxonomy	Обзор Структура Поиск Вставить Очистить Удалить	1	InnoDB	utf8mb4_unicode_520_ci	48,0 Киб	-
<input type="checkbox"/> wp_usermeta	Обзор Структура Поиск Вставить Очистить Удалить	17	InnoDB	utf8mb4_unicode_520_ci	48,0 Киб	-
<input type="checkbox"/> wp_users	Обзор Структура Поиск Вставить Очистить Удалить	1	InnoDB	utf8mb4_unicode_520_ci	64,0 Киб	-
12 таблиц	Всего	181	InnoDB	utf8mb4_general_ci	2,1 Миб	0 Байт

↑ ☐ Отметить все С отмеченными: ▼

Рисунок 11 – Созданная базы данных weblab3

После этого необходимо перейти по адресу <http://localhost/wordpress>. Там появится форма, в которой нужно будет указать имя БД, имя пользователя, пароль и некоторые другие параметры. Далее откроется окно форма для заполнения информации о создаваемом сайте.

WordPress › Установка

Добро пожаловать

Добро пожаловать в знаменитую пятиминутную установку WordPress! Просто заполните поля — и вперёд, к использованию самой мощной и гибкой персональной платформы для публикаций в мире!

Требуется информация

Пожалуйста, укажите следующую информацию. Не переживайте, потом вы всегда сможете изменить эти настройки.

Название сайта

Имя пользователя

Имя пользователя может содержать только латинские буквы, пробелы, подчёркивания, дефисы, точки и символ @.

Пароль [Скрыть](#)

Очень слабый

Важно: Этот пароль понадобится вам для входа. Сохраните его в надёжном месте.

Подтвердите пароль ☒ Разрешить использование слабого пароля.

Ваш e-mail

Внимательно проверьте адрес электронной почты, перед тем как продолжить.

Видимость для поисковых систем ☐ Попросить поисковые системы не индексировать сайт

Будет ли учитываться этот запрос — зависит от поисковых систем.

[Установить WordPress](#)

Рисунок 12 – Установка WordPress

После успешной установки мы попадем в консоль разработчика только что созданного сайта.

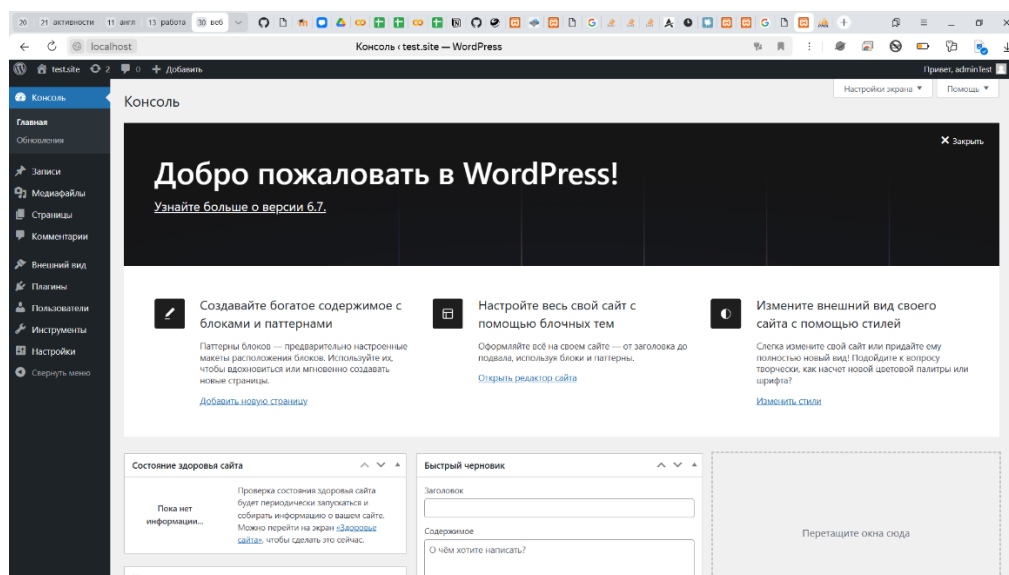


Рисунок 13 – Открывшаяся консоль разработчика

Далее необходимо сделать так, чтобы при переходе по ссылке <http://test.site>, открывался сайт.

Во-первых, необходимо прописать виртуальные хосты. Для этого необходимо отредактировать файл `httpd-vhosts.conf`. Было прописано два хоста, чтобы при этом работал еще и `localhost`.

```
<VirtualHost _default_:80>
    DocumentRoot "D:/XAMPP/htdocs"
    ServerName localhost
    ServerAlias localhost
    <Directory "D:/XAMPP/htdocs">
        Options +Indexes +Includes +FollowSymLinks +MultiViews
        AllowOverride All
        Require local
    </Directory>
    ErrorLog "logs/localhost.log"
    CustomLog "logs/localhost.log" common
</VirtualHost>

<VirtualHost *:80>
    DocumentRoot "D:/XAMPP/htdocs/wordpress"
    ServerName test.site
    ServerAlias test.site
    <Directory "D:/XAMPP/htdocs/wordpress">
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog "logs/test.log"
    CustomLog "logs/test.log" common
</VirtualHost>
```

Рисунок 14 – Создание виртуальных портов

Во-вторых, необходимо добавить в файл `hosts` в папке `C:\Windows\System32\drivers\etc` следующее описание адреса.

```
# End of section
127.0.0.1 test.site
```

Рисунок 15 – Редактирование файла `hosts`

Далее необходимо произвести манипуляции с базой данных `weblab3` на листе `wp_options`. Изменить переменные `siteurl` и `home` на <http://test.site>.

			option_id	option_name	option_value	autoload
<input type="checkbox"/>	Изменить	Копировать	Удалить	1	cron	a:9:{i:1732025843;a:1:{s:16:"wp_version_check";a:1... on
<input type="checkbox"/>	Изменить	Копировать	Удалить	2	siteurl	http://test.site on
<input type="checkbox"/>	Изменить	Копировать	Удалить	3	home	http://test.site on
<input type="checkbox"/>	Изменить	Копировать	Удалить	4	blogname	test.site on

Рисунок 16 – Изменение переменных в базе данных

После этого переходим по адресу <http://test.site>. После чего появляется приветственная страница.

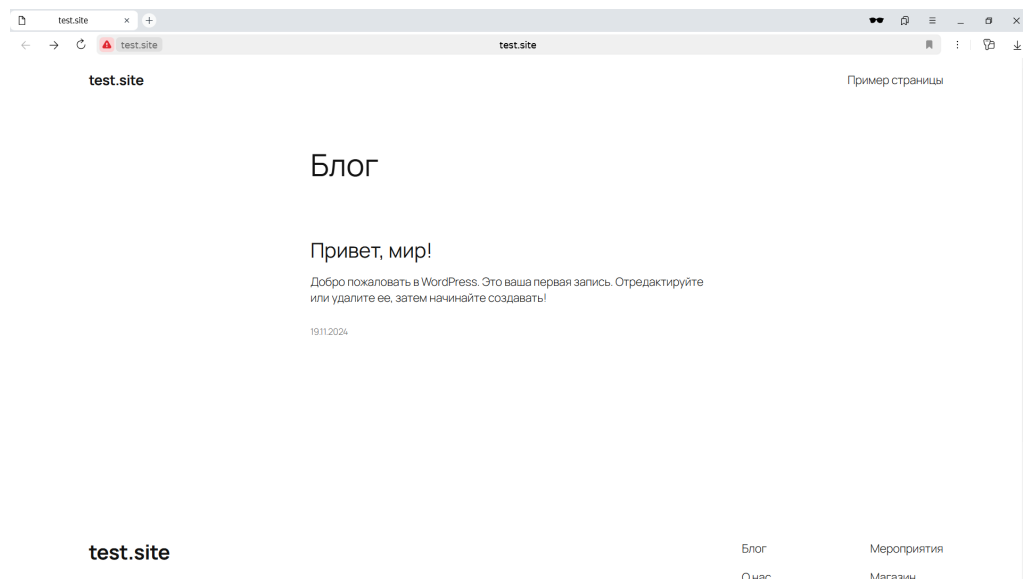


Рисунок 17 – Приветственная страница WordPress

Вывод

В ходе выполнения лабораторной работы цель была достигнута. Было произведено знакомство с PHP-скриптами и локальным сервером XAMPP.

В течение выполнения работы были изучены использование параллельных и последовательных задач Gulp, после этого были изучены post- и get-запросы с помощью PHP-скриптов, был написан сайт с приемом и отображением обратной связи. Также, был установлен инструментарий для локальных серверов XAMPP, с помощью которого была произведена работа с базой данных, скачан WordPress, произведена его установка и настроен переход на сайт по ссылке <http://test.site>.