**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**ITMO University**

Факультет «Факультет инфокоммуникационных технологий»

Направление подготовки «Инфокоммуникационные технологии и системы связи»

Отчет по дисциплине: «Web-программирование»

Практическая работа №3

Выполнила:

Колтунова Полина Владимировна

Группа №K3320

Проверила:

Марченко Елена Вадимовна

Санкт-Петербург

2024

**ВВЕДЕНИЕ**

Целью работы является изучение gulp, форм, php скриптов и Wordpress.

Задачи:

1. Настроить gulp: а) создать два таска – настроить на последовательное и параллельное выполнение; б) настроить отображение файлов проекта в браузере и автоматическую перезарузку при изменении одного из контролируемых файлов проекта.

2. Создать форму для отправки информации по обратной связи от пользователя сайта – передает информацию о себе: имя, фамилия, электронная почта, поле с обратной связью, создать радиокнопки и чекбоксы. Разработать файл с формой и php скрипт по образцу.

3. Установить инструментарий для отладки проектов. Установить движок с портала wordpress. Настроить портал <http://test.site>, чтобы при вводе данного адреса, отвечал настроенный портал.

**ГЛАВА 1. ХОД РАБОТЫ**

**Задание 1. Настройка gulp и создание тасков**

**Пункт а.**

Для последовательного выполнения использована функция series(), которая параметрами принимает любое количество задач и выполняет их по очереди в порядке перечисления (рис.1).

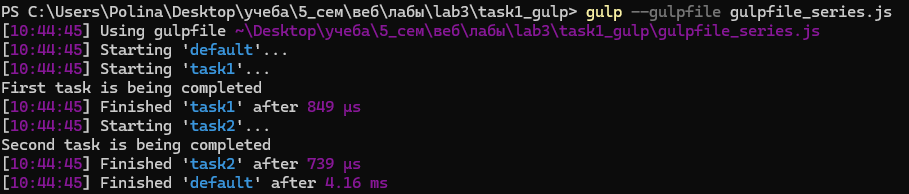


Рисунок 1 - Последовательно выполнение

Для параллельного выполнения использована функция parallel(), которая параллельно выполняет задачи, передаваемые в качестве параметров (рис.2).

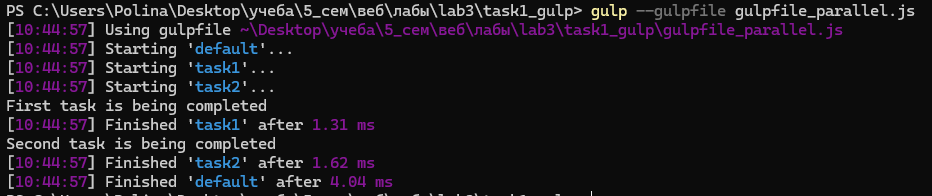


Рисунок 2 - Параллельное выполнение

**Пункт б.**

С помощью команды npm i browser-sync был установлен инструмент автоматизации BrowserSync, который синхронизирует изменения в коде и автоматически обновляет страницу в браузере, что позволяет видеть результаты сразу после внесения изменений.

Код файла gulpfile\_reload.js запускает сервер **BrowserSync**, который следит за изменениями в файлах проекта с помощью функции watchFiles(). Когда изменения происходят, вызывается функция reload(), которая перезагружает страницу в браузере. Задачи выполняются последовательно с помощью функции series(), которая последовательно запускает сервер, а затем начинается наблюдение за файлами.

Процесс работы gilp с BrowserSync на рис.3.

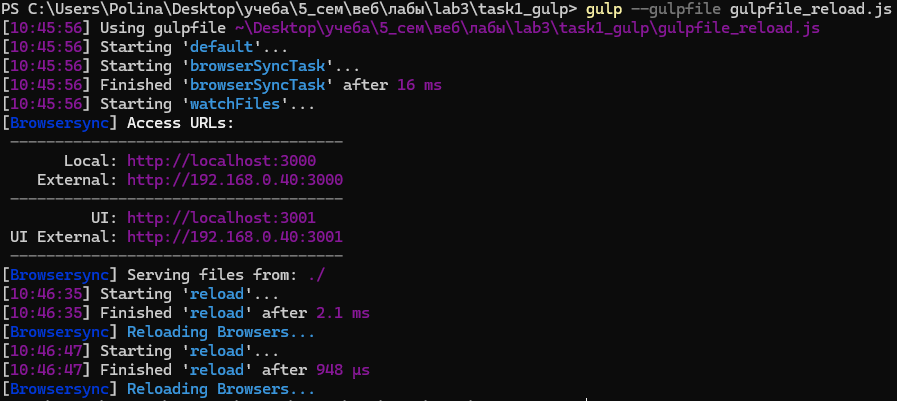


Рисунок 3 – Запуск задачи

Страница до изменений в файлах проекта (рис.4):



Рисунок 4 - Вид страницы до изменений

Страница после изменений в файлах проекта (рис.5):

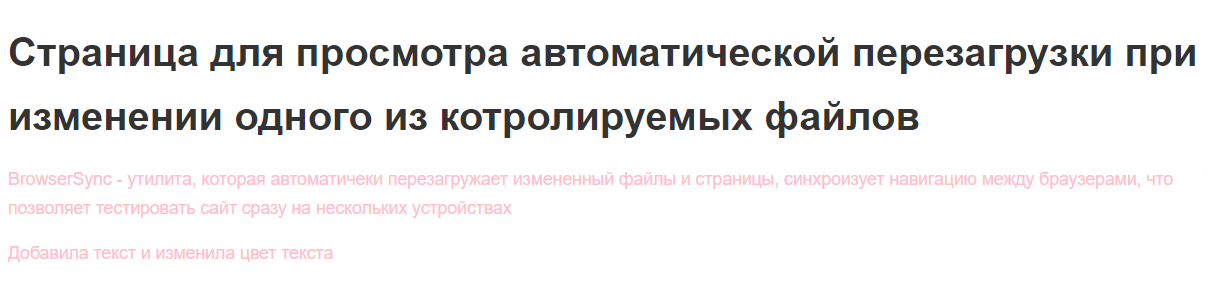


Рисунок 5 - Вид страницы после изменений

**Задание 2. Создание формы**

Был создан файл index.html (рис.6), содержащий форму обратной связи, где пользователь вводит своё имя, фамилию, электронную почту, сообщение, выбирает варианты ответа на вопросы, а затем отправляет данные через метод POST на сервер для обработки скриптом process.php (рис.7). В файле index.html подключается внешний файл стилей CSS styles.css, который используется для оформления и стилизации элементов на странице.

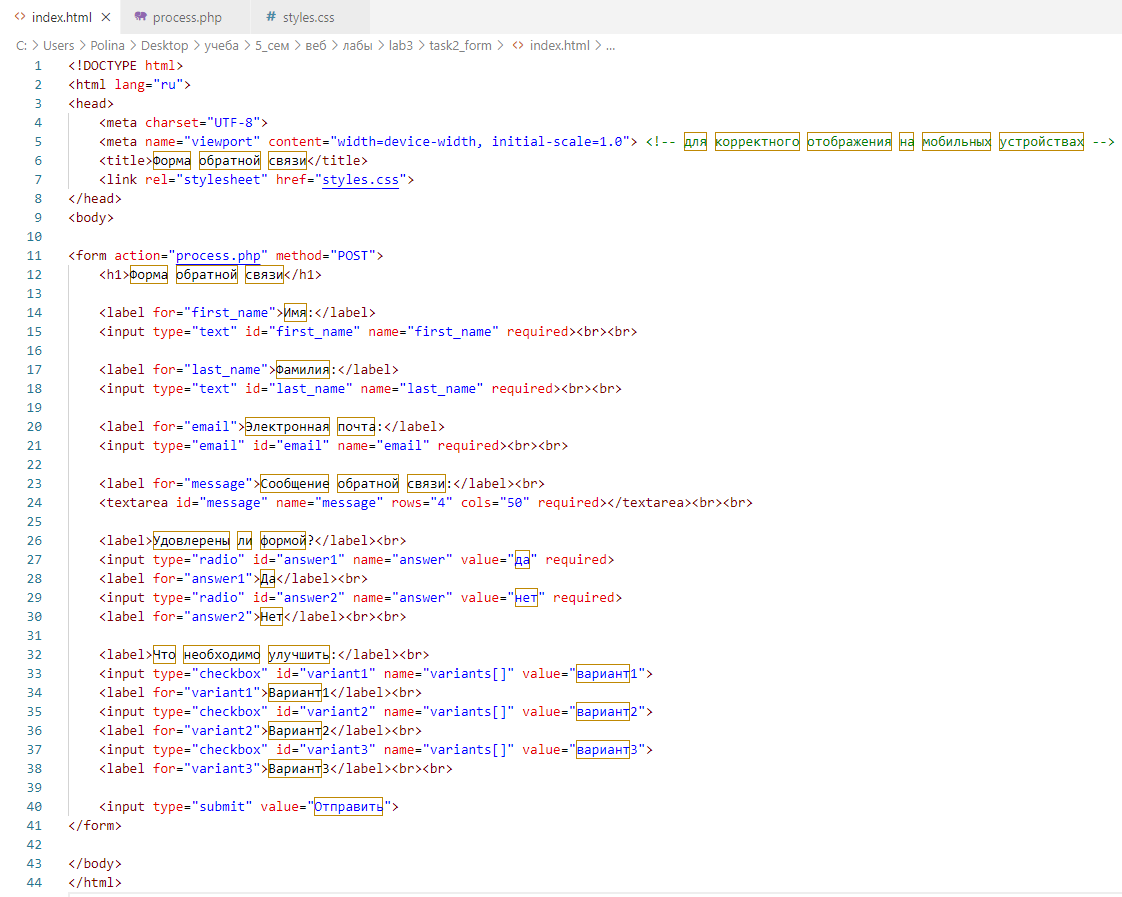
****

Рисунок 6 – Форма обратной связи

PHP предоставляет два основных метода для передачи данных между браузером и сервером: GET и POST.

Метод **GET** предназначен для запроса данных с сервера. Когда используется этот метод, параметры запроса передаются через URL в строке запроса, которая начинается с символа «?» и состоит из пар ключей и значений, разделенных знаком «&». Данные, передаваемые через него, отображаются в адресной строке браузера, а сам запрос может быть закэширован и сохранен в закладках. С помощью данного метода можно совершать поиск информации. Длина URL ограничена (обычно до 2000 символов), что накладывает ограничения на объем передаваемых данных.

Метод **POST** используется для отправки данных на сервер. Данные передаются в теле HTTP-запроса, что делает их невидимыми для пользователя в адресной строке. POST не имеет строгих ограничений на объем передаваемых данных, что позволяет использовать его для отправки больших файлов. Запросы, выполненные с использованием POST, не кэшируются и не сохраняются в закладках, т.к. они изменяют данные на сервере, п=а повторный отправленный запрос может привести к нежелательным последствиям (например, к созданию дубликатов)



Рисунок 7 – Обработка формы

#### Для установки php с сайта <https://windows.php.net/download> был скачан интерпретатор php Thread Safe. После распаковки архива была проверена установка (рис.8).

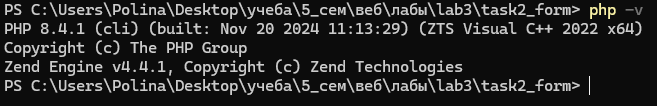


Рисунок 8 - Проверка установки php

Запуск встроенного php сервера с помощью команды php -S localhost:8000 (рис.9):



Рисунок 9 - Запуск встроенного php сервера

Для проверки работы в строке браузера был введен адрес <http://localhost:8000>, после чего открылась форма (рис.10).

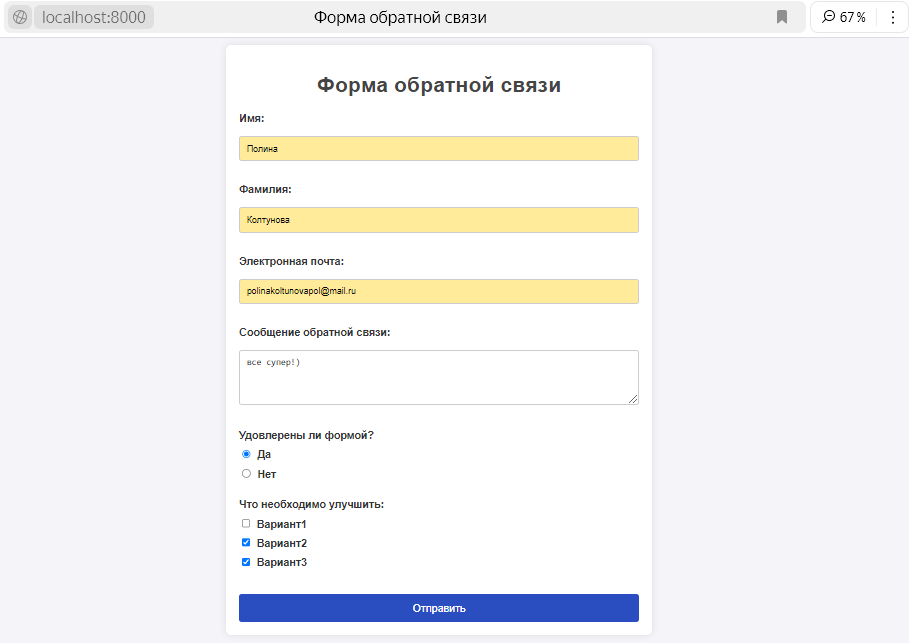


Рисунок 10 - Форма обратной связи

После заполнения и отправки формы, php скрипт обрабатывает данные формы и выводится результат (рис.11)

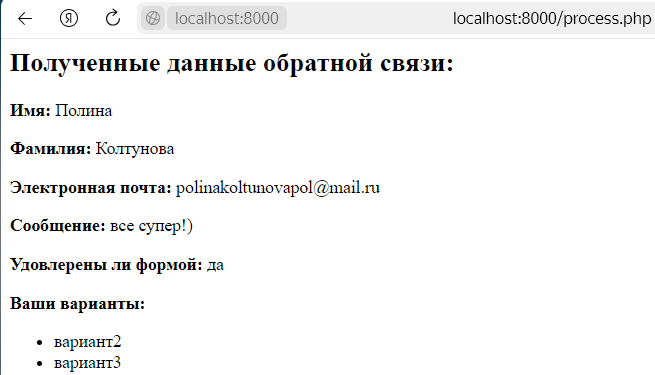


Рисунок 11 - Результат отправки формы

**Задание 3. Работа с WordPress**

1. **Установка инструментария для локального сервера**

Был установлен и настроен XAMPP для кроссплатформенной сборки локального веб-сервера с официального сайта <https://www.apachefriends.org/download.html> и скачана версия для Windows.

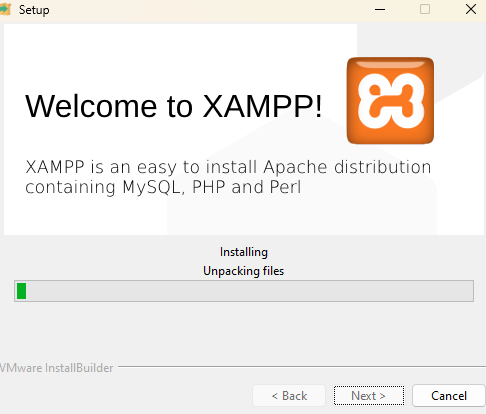


Рисунок 12 - Установка XAMPP

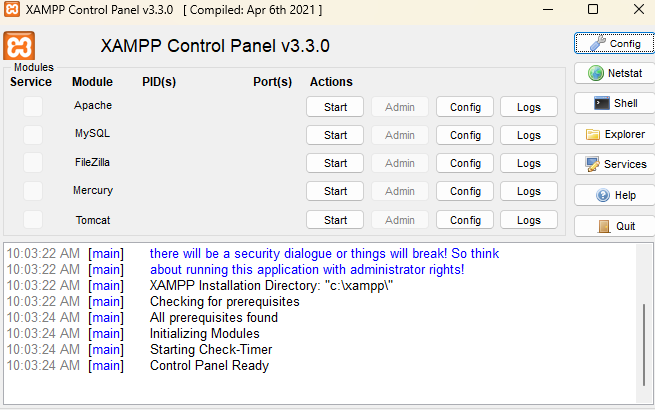


Рисунок 13 - Панель управления XAMPP

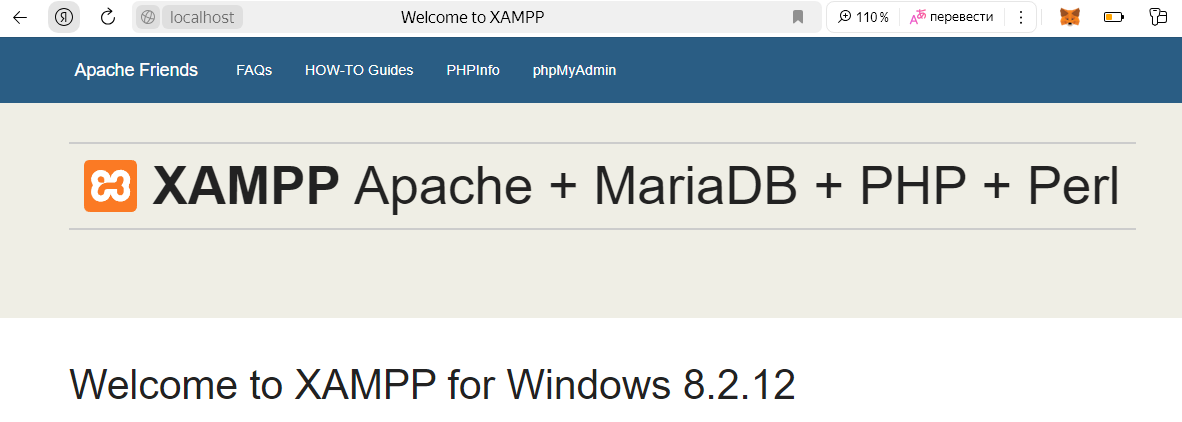


Рисунок 14 - Проверка запуска XAMPP

1. **Скачивание и установка WordPress**

С официального сайта <https://ru.wordpress.org/> была скачана последняя версия. Скачанный файл был разархивирован в каталог C:\xampp\htdocs и переименован в папку test.site.

Затем в браузере, перейдя на http://localhost/phpmyadmin, была создана база данных test\_site (рис.15).

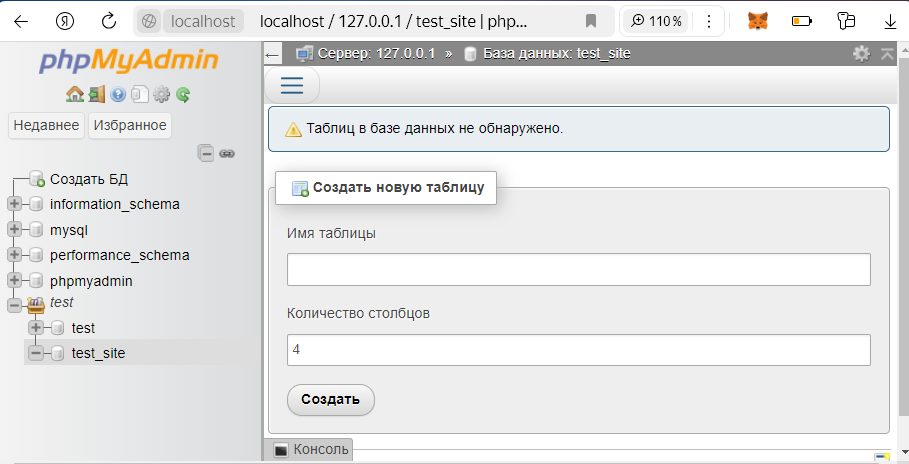


Рисунок 15 - Создание базы данных test\_site

Далее на <http://localhost/test.site>, был настроен файл конфигурации с помощью внесения данных бд (название: test\_site, имя пользователя: root, пароль: без, сервер бд: localhost) и закончена установка (рис. 16-18).

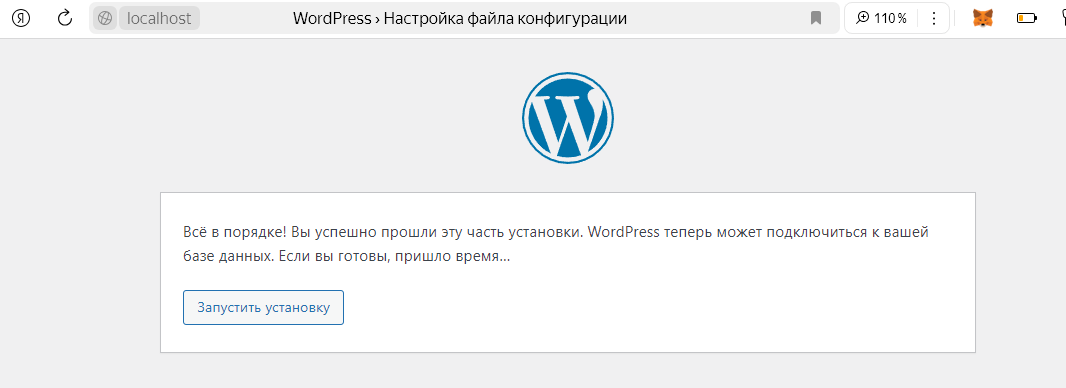


Рисунок 16 - настройка файла конфигурации

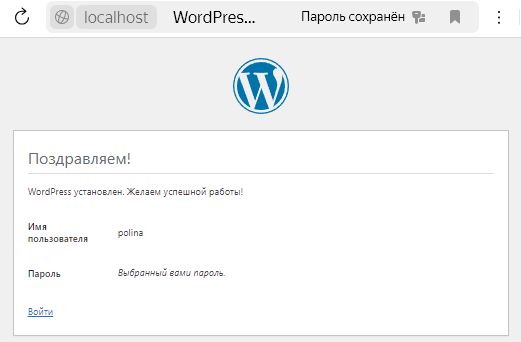


Рисунок 17 - Успешная установка WordPress

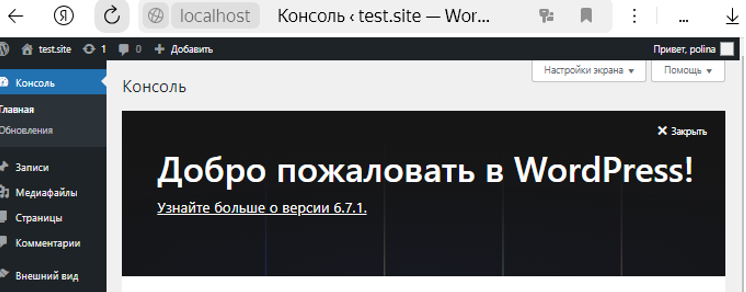


Рисунок 18 - Вход в WordPress

**3. Настройка локального домена** <http://test.site>

Был назначен IP-адрес 127.0.0.1 для домена test.site. Для это был отредактирован файл hosts, содержащий информацию о ip-адресах и их доменах с правами администратора (путь к файлу: C:\Windows\System32\drivers\etc\hosts). В него добавлена строка 127.0.0.1 test.site (рис.19).

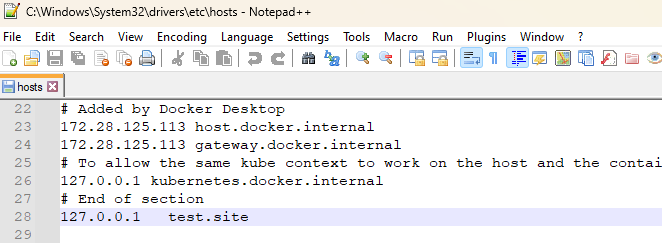


Рисунок 19 - Редактирование файла hosts

Настройка Apache (рис.20). В файле конфигурации Apache (путь к файлу: C:\xampp\apache\conf\extra\httpd-vhosts.conf) был добавлен виртуальный хост

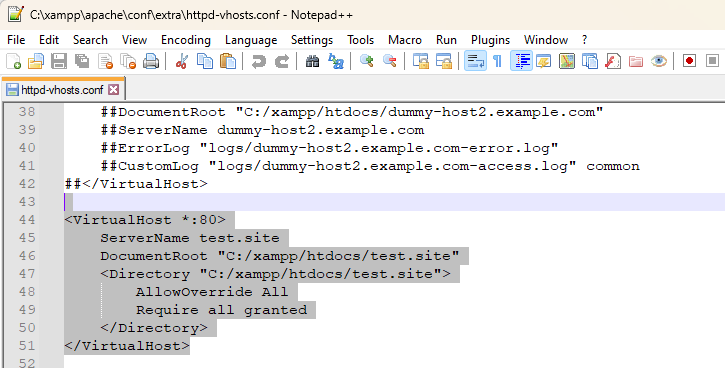


Рисунок 20 - Добавление виртуального хоста

После перезапуска Apache, в браузере был введен локальный домен <http://test.site>, и открыт настроенный портал WordPress (рис.21).

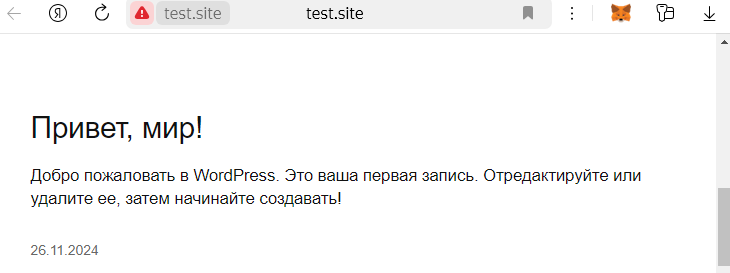


Рисунок 21 - Настроенный портал на WordPress

**ВЫВОД**

Цель была достигнута. Были получены навыки работы с gulp. Были созданы таски для последовательного и параллельного выполнения. Был изучен сервер **BrowserSync, с помощью которого настроена автоматическая перезагрузка при изменении одного из контролируемых файлов проекта. Была создана форма для отправки информации обратной связи, которая обрабатывается php скриптом. Установлен инструментарий XAMMP для локального сервера, скачан и установлен WordPress. Был настроен портал** http://test.site**, что при вводе данного адреса, отвечает настроенный портал.**