Санкт-Петербургский Национальный Исследовательский Университет

Информационных Технологий, Механики и Оптики

Факультет Инфокоммуникационных Технологий

**Web-программирование**

**Лабораторная работа 3**

Выполнил

Носов А.Д.

Проверила

Марченко Е. В.

Санкт-Петербург, 2024

**Цель:** научиться работать с gulp, wordpress, cоздать форму для отправки информации по обратной связи от пользователя сайта.

**Задачи:**

* Научиться работать с gulp: создать два таска и настроить на последовательное и параллельное выполнение; настроить отображение файлов проекта в браузере и автоматическую перезагрузку при изменении контролируемых файлов проекта.
* Создать форму для отправки информации по обратной связи от пользователя сайта согласно заданию.
* Настроить локальный сервер и установить wordpress.

**Ход работы**

**Задание 1**

С помощью команды npm install --global gulp-cli установим Gulp глобально.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 1 - Установка Gulp глобально

Затем инициализируем проект и установим Gulp как зависимость.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2 – Команда npm init -y

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 3 – Команда npm install --save-dev gulp

Затем установим пакет browsesync через команду

npm install browser-sync gulp –save-dev.

Cоздадим файл gulpfile.js в корне проекта, он определяет задачи, которые будет выполнять gulp. Создадим 1 таск, который будет обновлять изменения в файле .html с помощью browsesync.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 4 – Задачи для gulp

Проверим успешность выполнения скрипта.

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Рисунок 5 – Файл ex.html до изменения

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 6 – Запуск скрипта

На скриншоте видно, что при изменении файла страница перегружается.

Теперь посмотрим на последовательное и параллельное выполнение.

Cоздадим файл gulpfile.js в корне проекта, он определяет задачи, которые будет выполнять gulp. Создадим 2 таски, которые будут выводить в консоль числа.Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 7 – Таски вывода чисел в консоль

Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение, Графическое программное обеспечение

Автоматически созданное описание

Рисунок 8 – Последовательное выполнение тасок.

Изображение выглядит как снимок экрана, текст, Мультимедийное программное обеспечение

Автоматически созданное описание

Рисунок 9 – Параллельное выполнение тасок.

На скриншотах видно, что, хоть таски и запускаются параллельно, но все равно числа выводятся последовательно из каждой таски, так как Node.js – однопоточная среда.

**Задание 2**

Была создана форма для отправки информации по обратной связи от пользователя сайта согласно условию – передает информацию о себе: имя, фамилия, электронная почта, поле с обратной связью, должны быть радиокнопки (не менее 2 шт.) и должны быть чекбоксы (не менее 3 шт.). Разработан html файл с формой и php скрипт для обработки методов get и post.

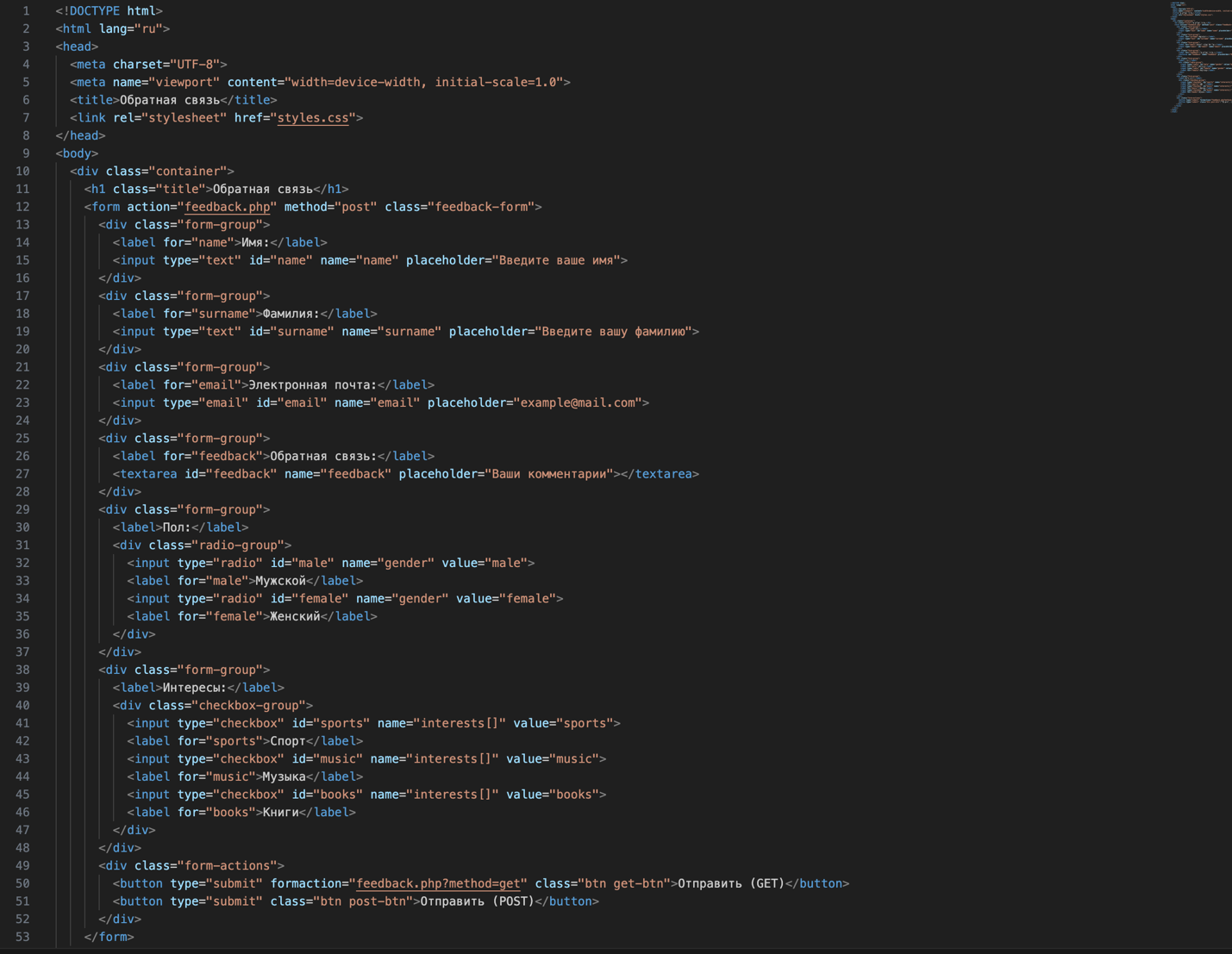


Рисунок 13 – Код html формы

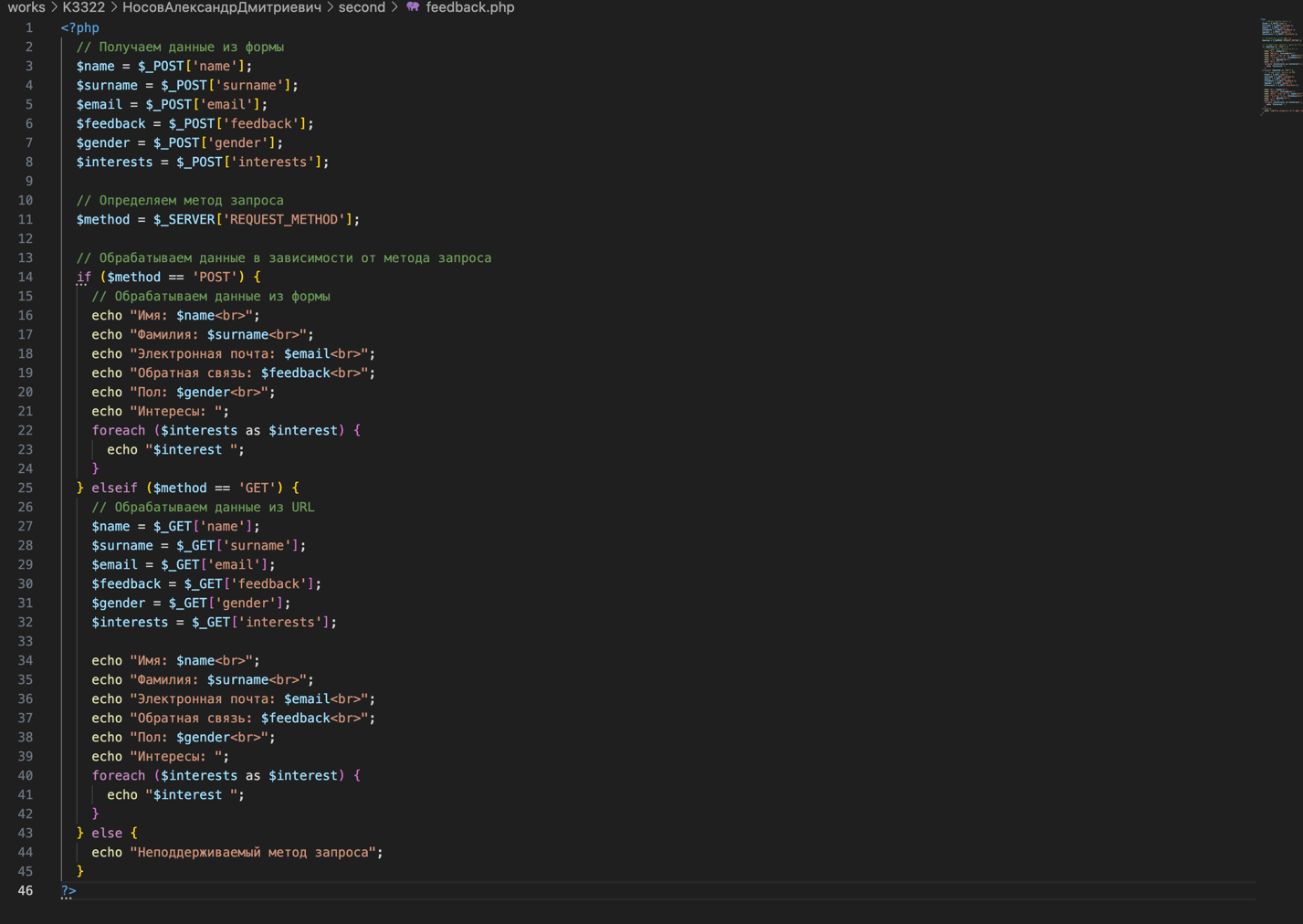


Рисунок 14 – php скрипт

Затем был установлен php с помощью Homebrew и запущен сервер.



Рисунок 15 – Запуск локального сервера

По ссылке открывается форма обратной связи.

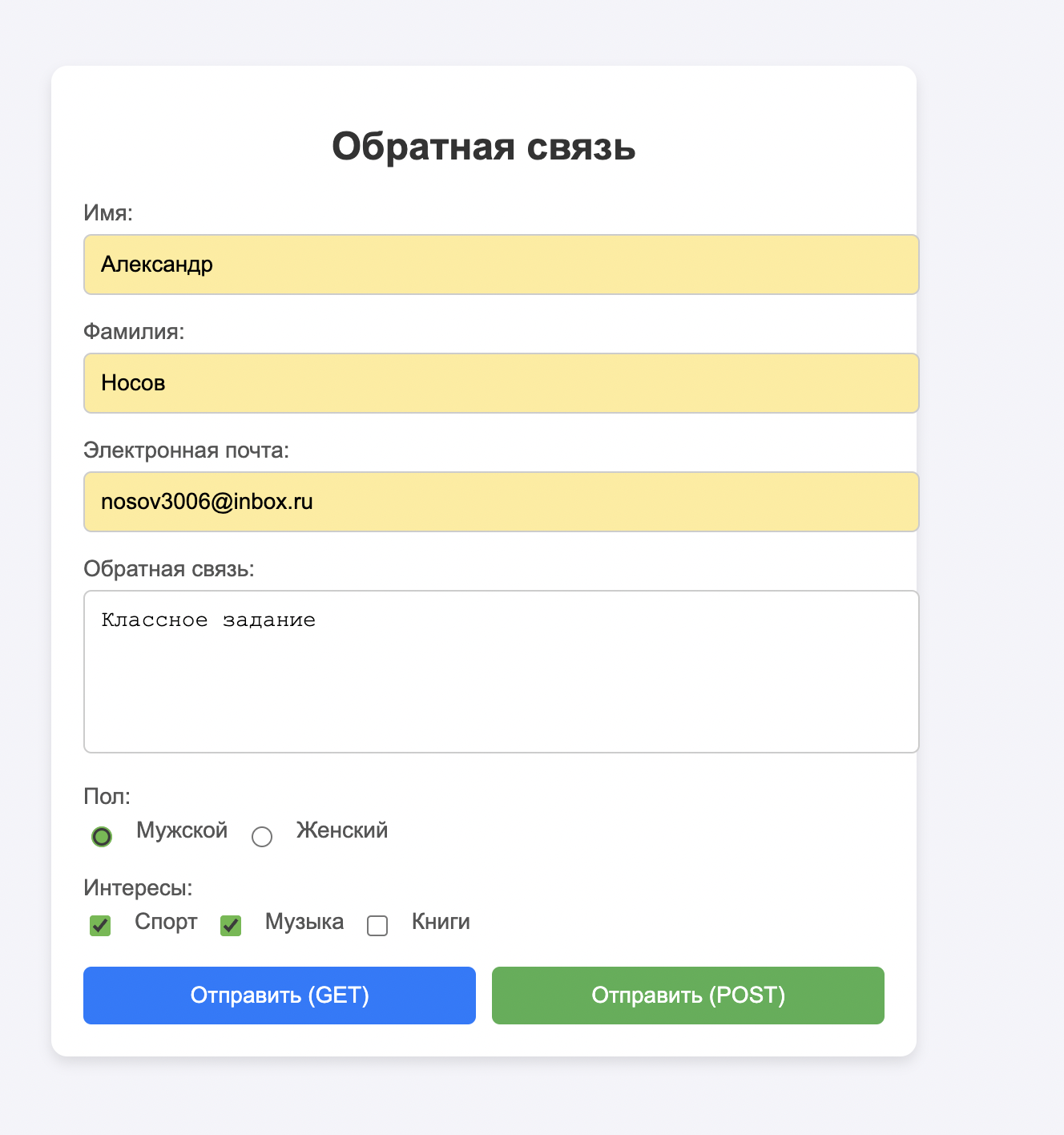


Рисунок 16 – Форма обратной связи в браузере

Заполняем форму и отправляем сначала методом POST.

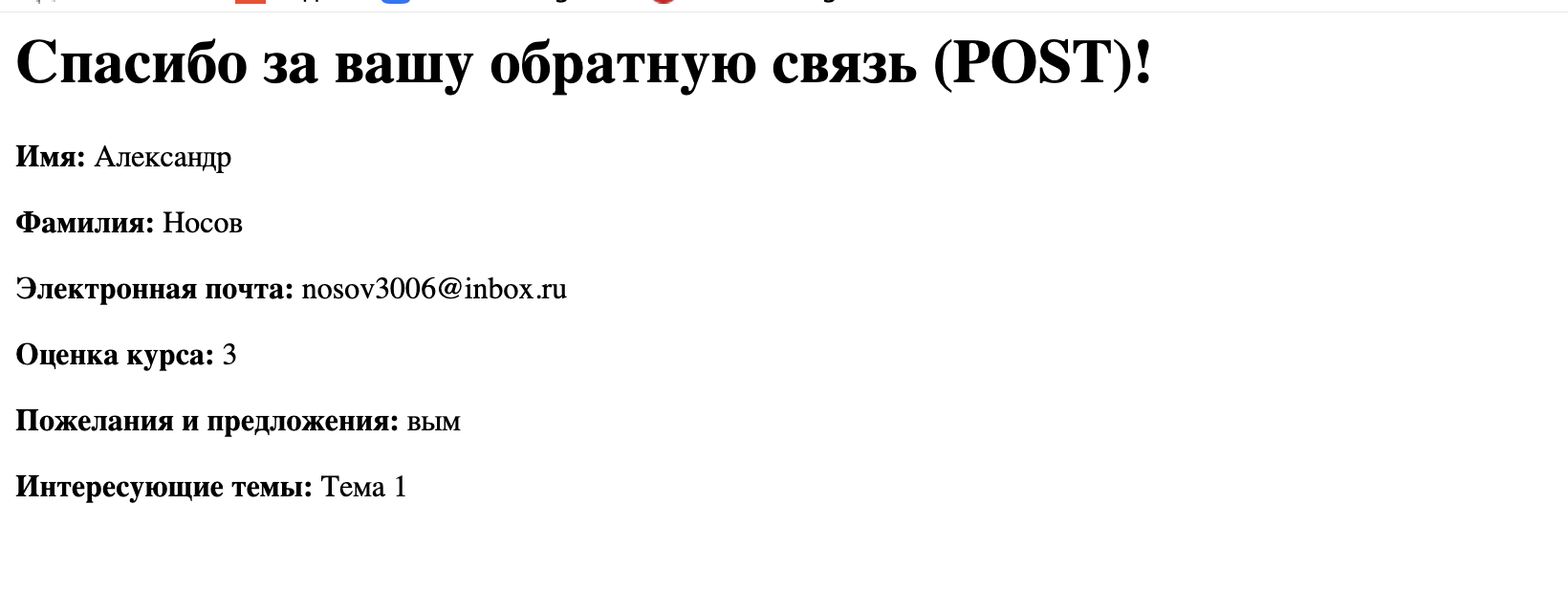


Рисунок 17 – Результат отправки методом POST

При отправке формы методом POST данные передаются на сервер в теле HTTP-запроса, а не через URL, что делает их невидимыми в адресной строке. Сервер принимает данные из массива $\_POST, обрабатывает их и возвращает ответ сообщение об успешной отправке. В отличие от метода GET, POST подходит для передачи больших объемов информации, например, длинных текстов или файлов и более безопасен, так как данные не отображаются в URL и не сохраняются в истории браузера.

Теперь отправим методом GET.

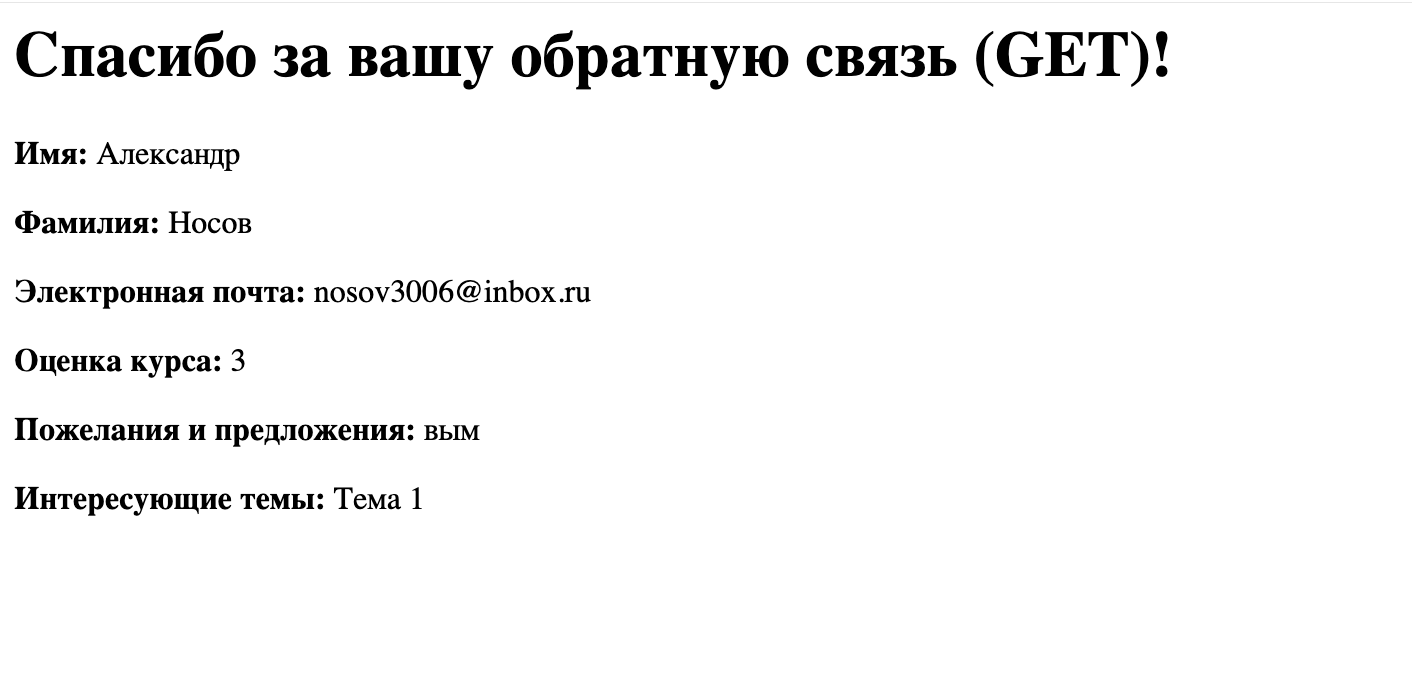


Рисунок 18 – Результат отправки методом GET

При отправке формы методом GET данные передаются на сервер через параметры в URL-адресе, которые видны в адресной строке после знака ?. Сервер принимает эти данные через массив $\_GET, обрабатывает их и возвращает ответ, отображая отправленную информацию. GET лучше подходит для запросов, которые не изменяют данные на сервере, например, поисковые запросы, и позволяет легко делиться URL с другими. В отличие от метода POST, GET ограничен длиной URL и менее безопасен, так как данные остаются видимыми в браузере и могут быть сохранены в истории.

**Задание 3**

Для установки WordPress будем использовать MAMP (Macintosh, Apache, MySQL, PHP) — это локальный серверный инструментарий, который позволяет запускать веб-приложения на компьютере под macOS.

Сначала скачаем MAMP с официального сайта и установим на компьютер. Получаем папку со следующим содержимым.

Изображение выглядит как текст, число, программное обеспечение, снимок экрана

Автоматически созданное описание

Рисунок 19 – MAMP

Далее скачаем последнюю версию WordPress также с официального сайта и распакуем его в папке htdocs.

Изображение выглядит как текст, число, программное обеспечение, Шрифт

Автоматически созданное описание

Рисунок 20 – WordPress

Теперь запустим сервер. Папка htdocs будет корневым каталогом документов, в ней будут храниться все файлы для локального веб-сервера.

Изображение выглядит как текст, снимок экрана, веб-страница, программное обеспечение

Автоматически созданное описание

Рисунок 21 – Запуск сервера

Создаем базу данных MySQL testsite.

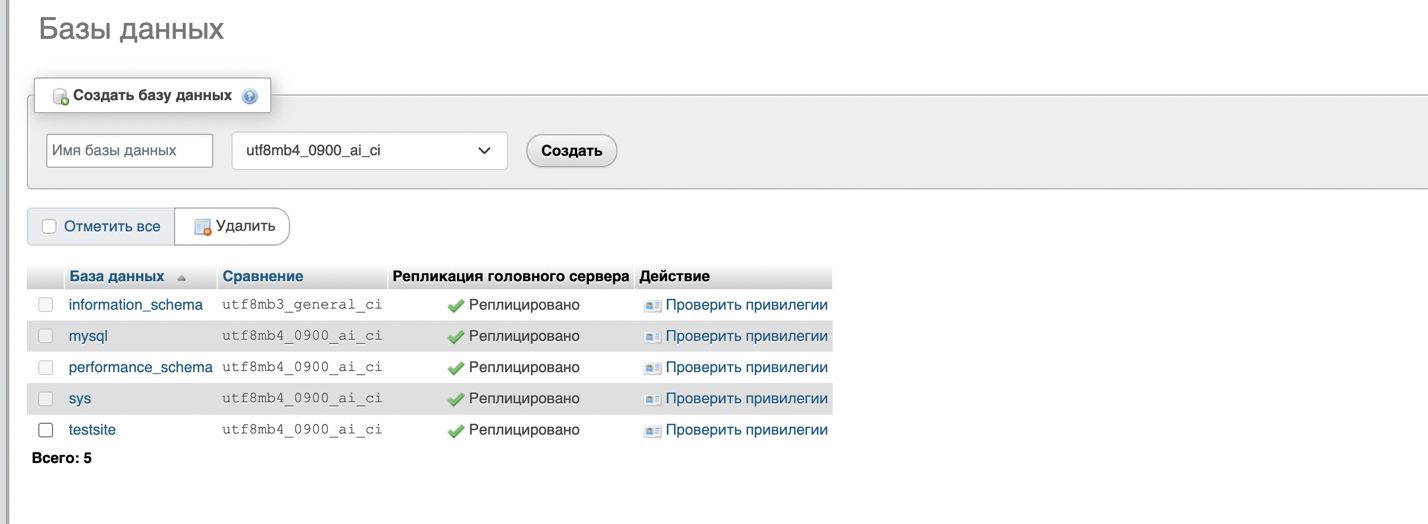


Рисунок 22 – Создание базы данных

После этого настроим WordPress. Переходим на сайт localhost:8888/wordpress, где wordpress – название папки. Здесь указываем имя созданной базы данных. Указываем имя сайта и наши личные данные.

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

Рисунок 23 – Настройка wordpress

Теперь можем зайти в панель управления wordpress <http://localhost:8888/wordpress/wp-admin>, где wordpress – название папки. А также открыть сайт на localhost.

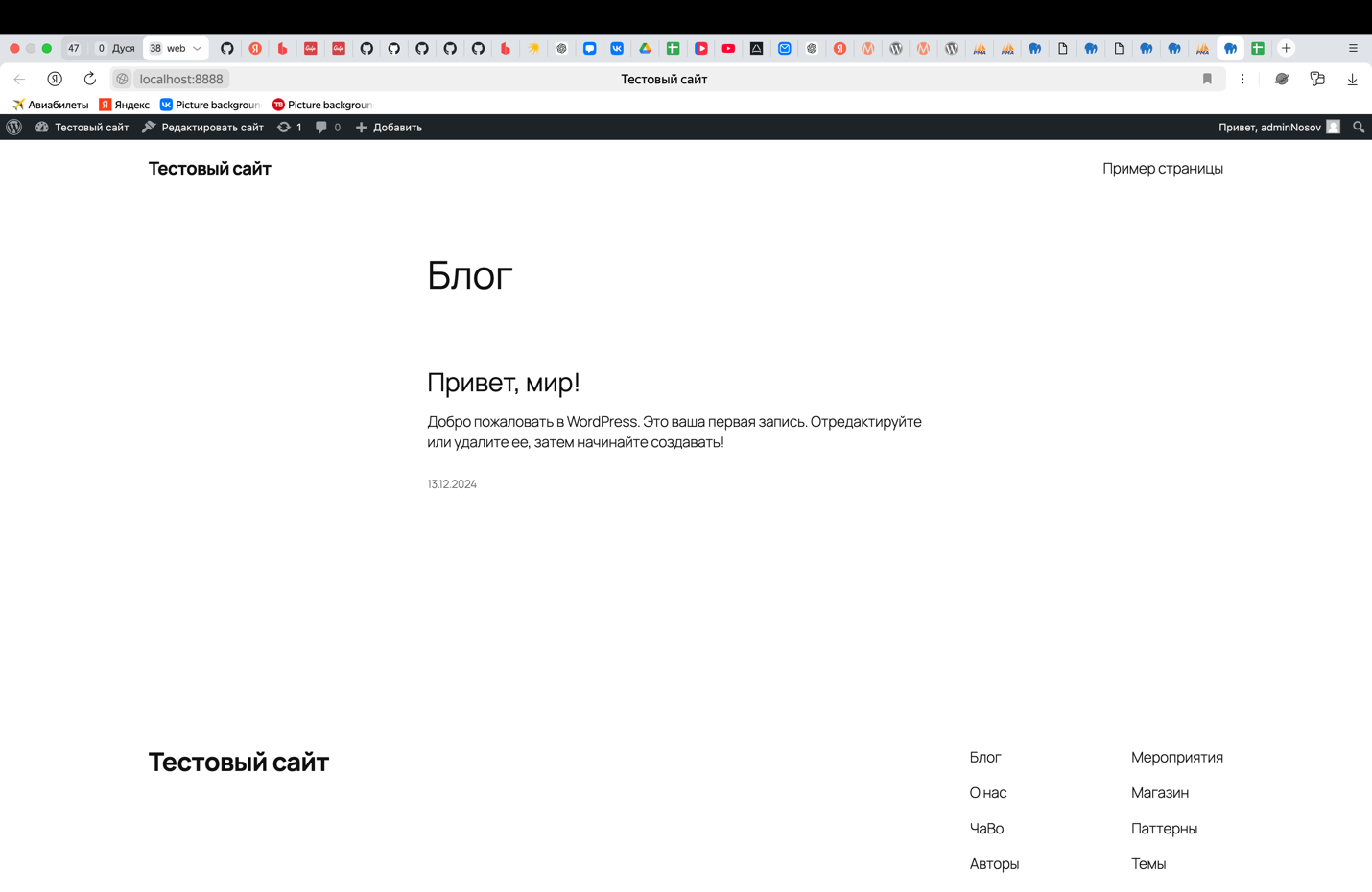


Рисунок 25 – Сайт

В терминале открываем файл hosts командой sudo nano /etc/hosts и добавляем следующую запись в конец файла.

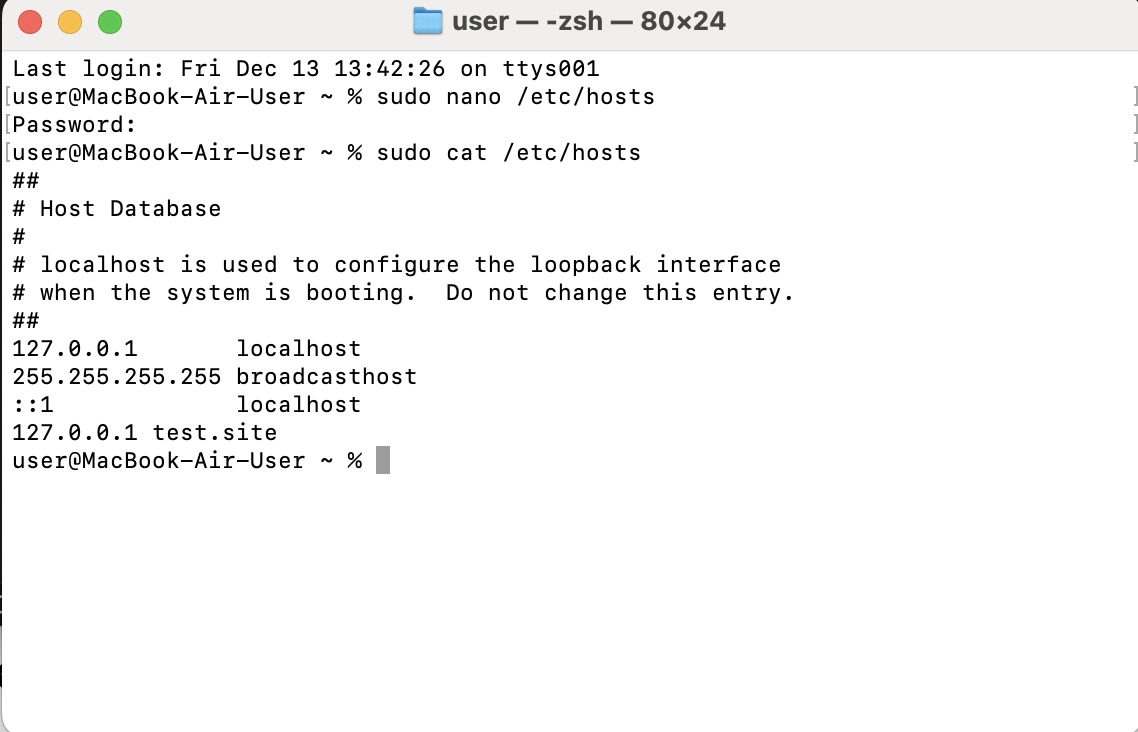


Рисунок 26 – Файл hosts

Был настроен виртуальный хост в Apache. Для этого был изменен файл Applications/MAMP/conf/apache/extra/httpd-vhost.conf.

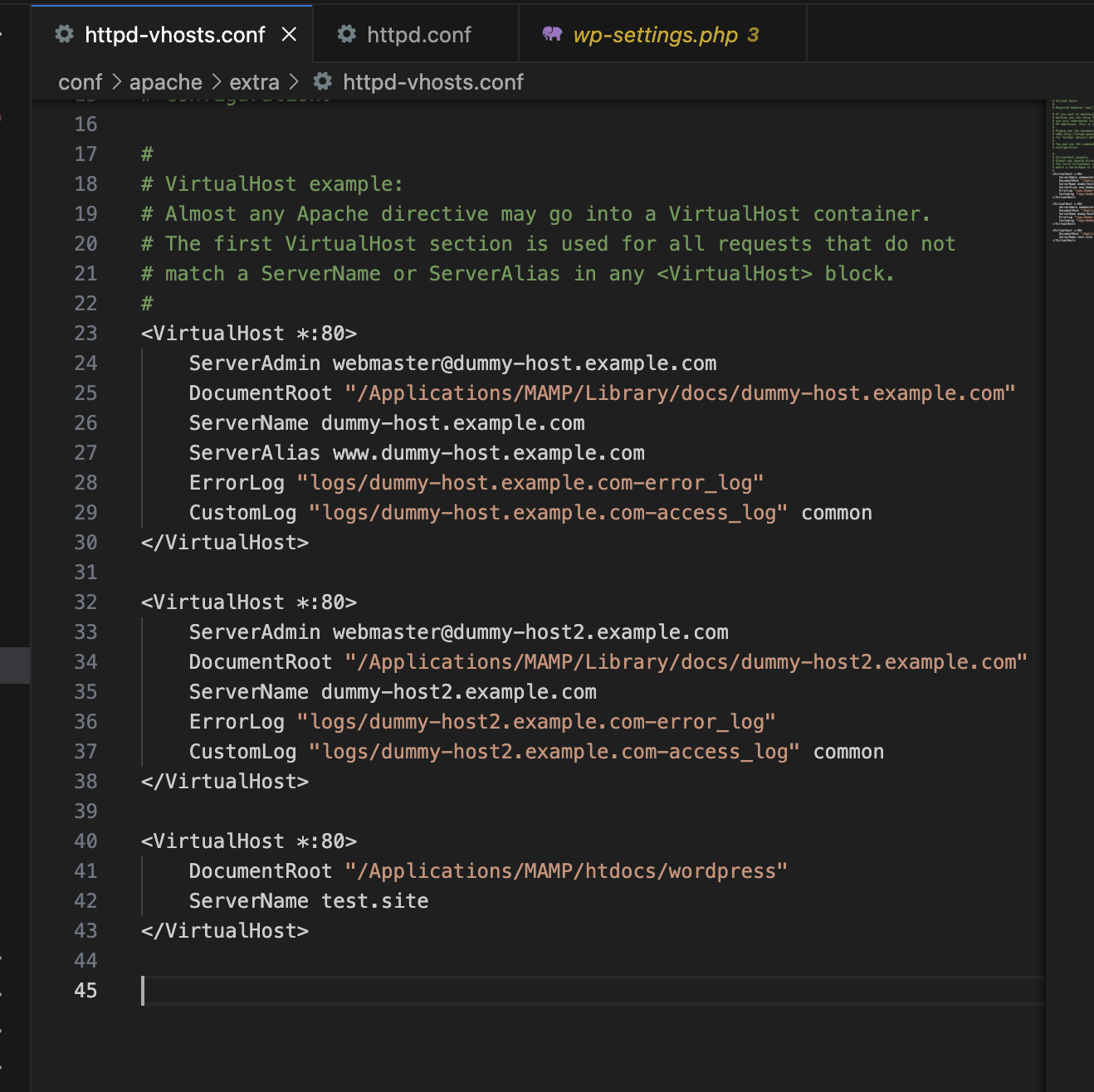


Рисунок 27 – Файл httpd-vhosts.conf

Раскомментируем следующие строки в файле Applications/MAMP/conf/apache/httpd.conf.

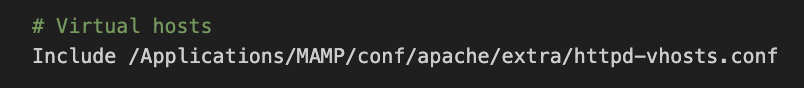


Рисунок 28 – Файл httpd.conf

Изображение выглядит как текст, Шрифт, белый, черный

Автоматически созданное описание

Рисунок 29 - Файл httpd.conf

В настройках wordpress заменим адрес сайта на <http://test.site> согласно условию задания.

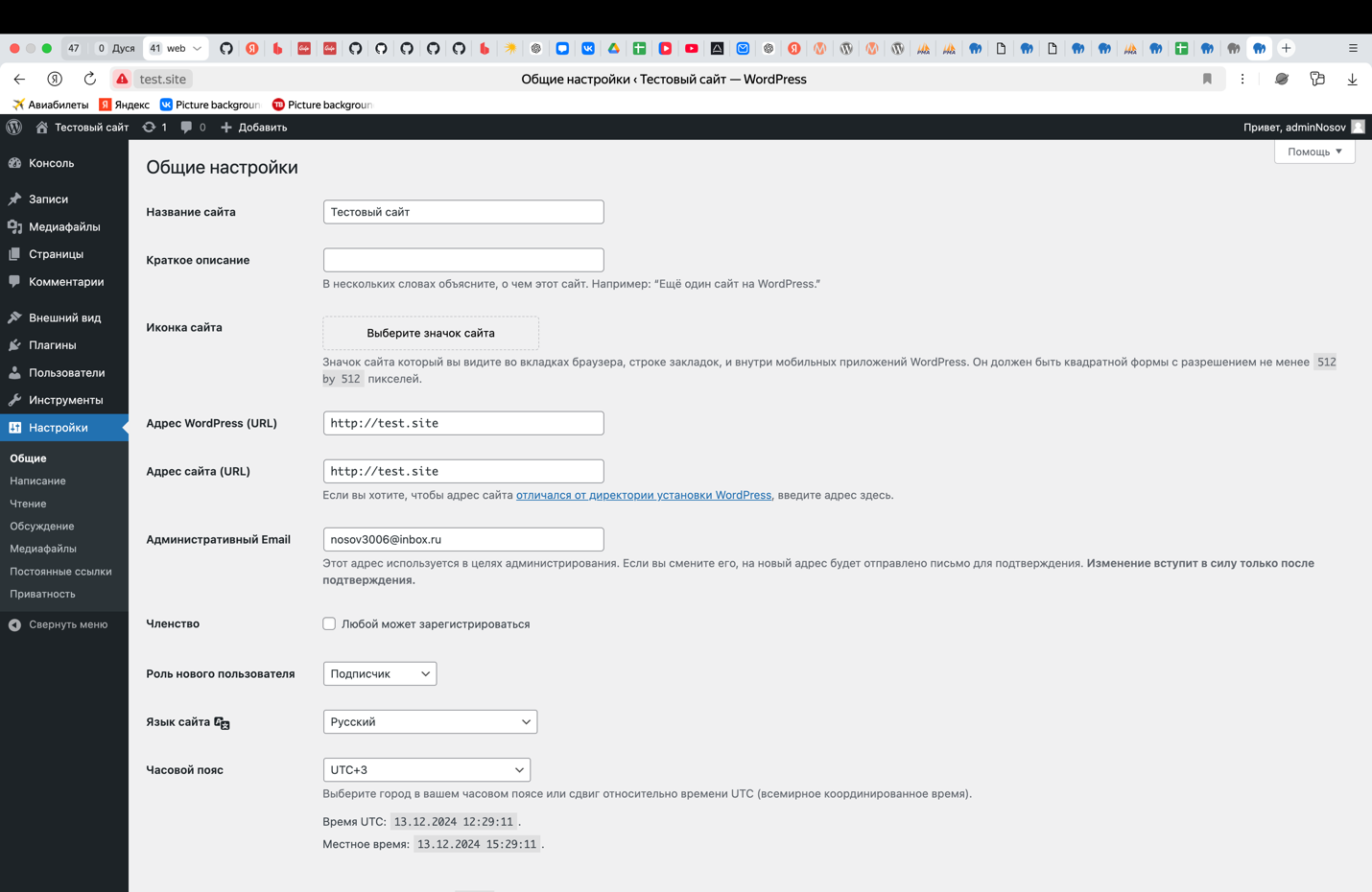


Рисунок 30 – Настройки wordpress

Также поставим новую тему для сайта. А при вводе <http://test.site> теперь успешно отображается сайт.

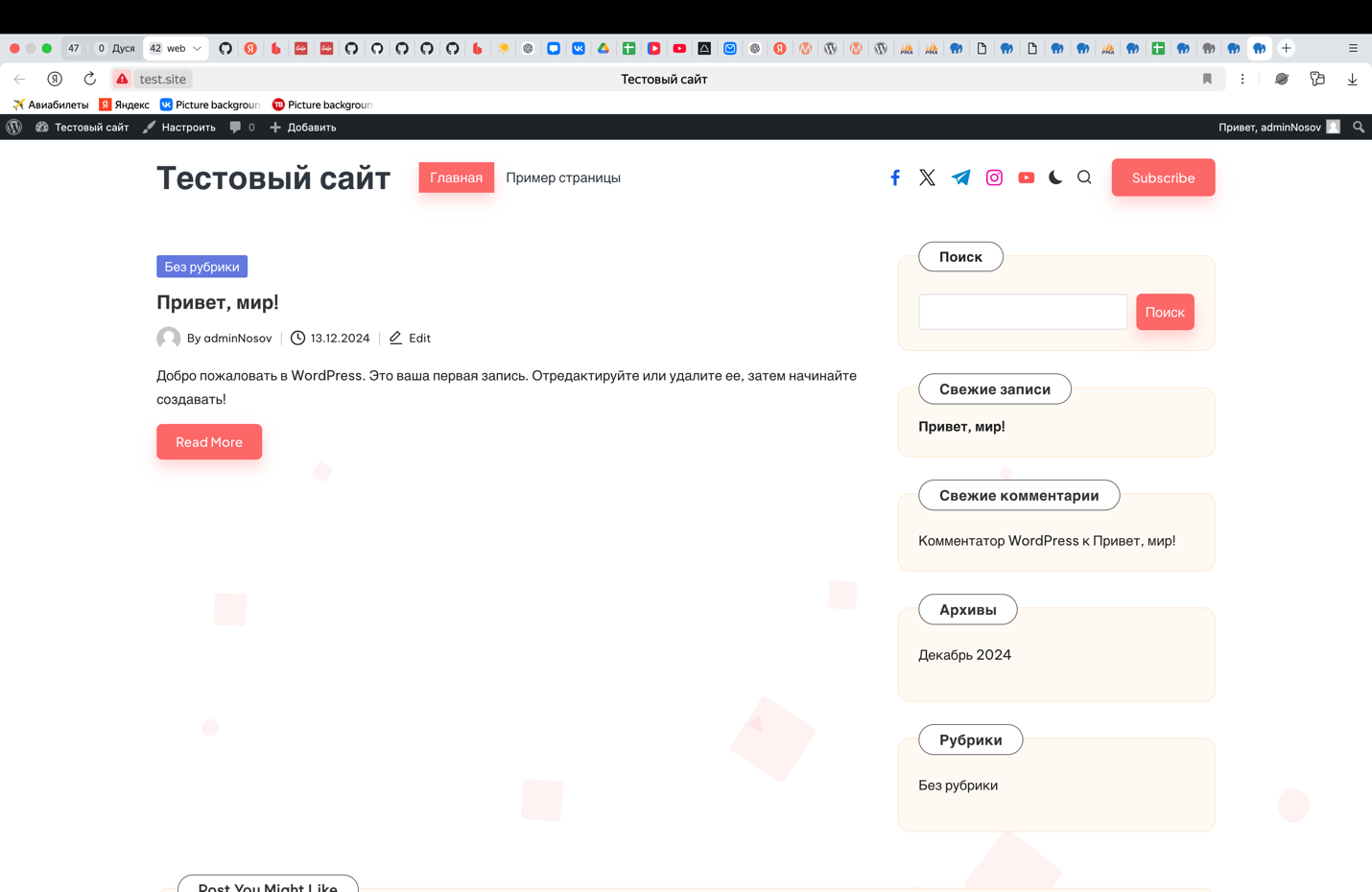


Рисунок 31 - Результат

**Вывод**

В результате выполнения лабораторной работы был изучен gulp, cоздана форма для отправки информации по обратной связи от пользователя сайта и настроен wordpress.