**Министерство науки и высшего образования Российской Федерации**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«Национальный исследовательский университет ИТМО»**

**(Университет ИТМО)**

Факультет инфокоммуникационных технологий

Отчет по дисциплине: **«Web-программирование»**

Лабораторная работа № 3

Выполнил: Волкович Кирилл

Группа: К3322

Проверила: Марченко Елена Вадимовна

Санкт-Петербург

2024

Содержание

[Введение. 3](#_Toc184589641)

[Часть 1. Последовательные и параллельные задачи в Gulp 4](#_Toc184589642)

[Часть 2. Форма обратной связи 6](#_Toc184589643)

[Часть 3. WordPress 9](#_Toc184589644)

[Заключение 12](#_Toc184589645)

# Введение.

**Цель**: изучить последовательные и параллельные задачи в Gulp, написать форму обратной связи, используя PHP-скрипты, а также при помощи Wordpress настроить переход на сайт test.site.

# Часть 1. Последовательные и параллельные задачи в Gulp

В этом задании необходимо было создать два таска и настроить их на последовательное и параллельное выполнение. Для этого были написаны две задачи: sum и dif, которые выполняют сложение и вычитание определенных переменных. После чего были созданы еще 2 задачи, которые при помощи series() и parallel() запускают либо последовательное выполнение задач sum и dif, либо же параллельное (Рисунок 1).

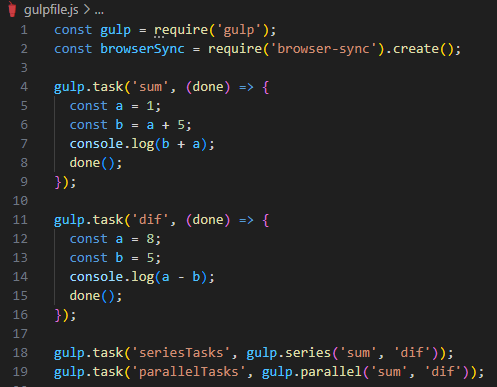


Рисунок 1 – Последовательные и параллельные задачи

После запуска задачи, отвечающей за последовательное выполнение, видим, что задачи выполняются по очереди, вторая задача ожидает выполнение первой и только тогда начинает выполняться (Рисунок 2).

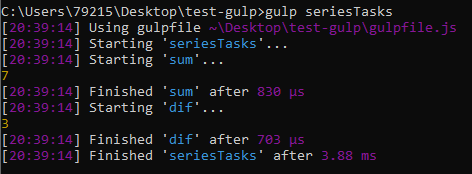


Рисунок 2 – Последовательное выполнение

При запуске задачи, отвечающей за параллельное выполнение, запускаются все задачи и начинают одновременно выполняться (Рисунок 3).

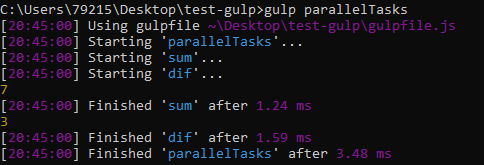


Рисунок 3 – Параллельное выполнение

Далее необходимо было создать задачу, при выполнении которой файлы проекта отображались в браузере, а также происходила бы автоматическая перезагрузка при изменении любого из контролируемых файлов проекта.

Была создана папка src, в которой созданы файлы index.html, script.js в подпапке js и style.css в подпапке css.

После этого создаются объект paths, который содержит пути к отслеживаемым файлам и задача serve, которая настраивает локальный сервер. Он будет обслуживать файлы из папки src. Далее прописываются методы watch, которые следят за изменениями в файлах. Если файл изменился, то выполняется команда browserSync.reload, чтобы обновить браузер. В итоге создается задача WEB, которая при помощи gulp.series('serve') выполняет задачу serve (Рисунок 4).



Рисунок 4 – Gulpfile

При выполнении задачи WEB будет открываться страница в браузере и обновляться при любом изменении контролируемых файлов (Рисунок 5).



Рисунок 5 – Страница в браузере

# Часть 2. Форма обратной связи

В данном задании было необходимо создать форму обратной связи, также необходимо было написать php-скрипт для корректной работы формы. Для начала был написан index.html с формой обратной связи (Рисунок 6). В которой необходимо заполнить имя, фамилию, почту, обратную связь, тип праздника при помощи радиокнопок, а также тип услуги при помощи чекбоксов. После заполнения форма отправляется при помощи кнопки (Рисунок 8).

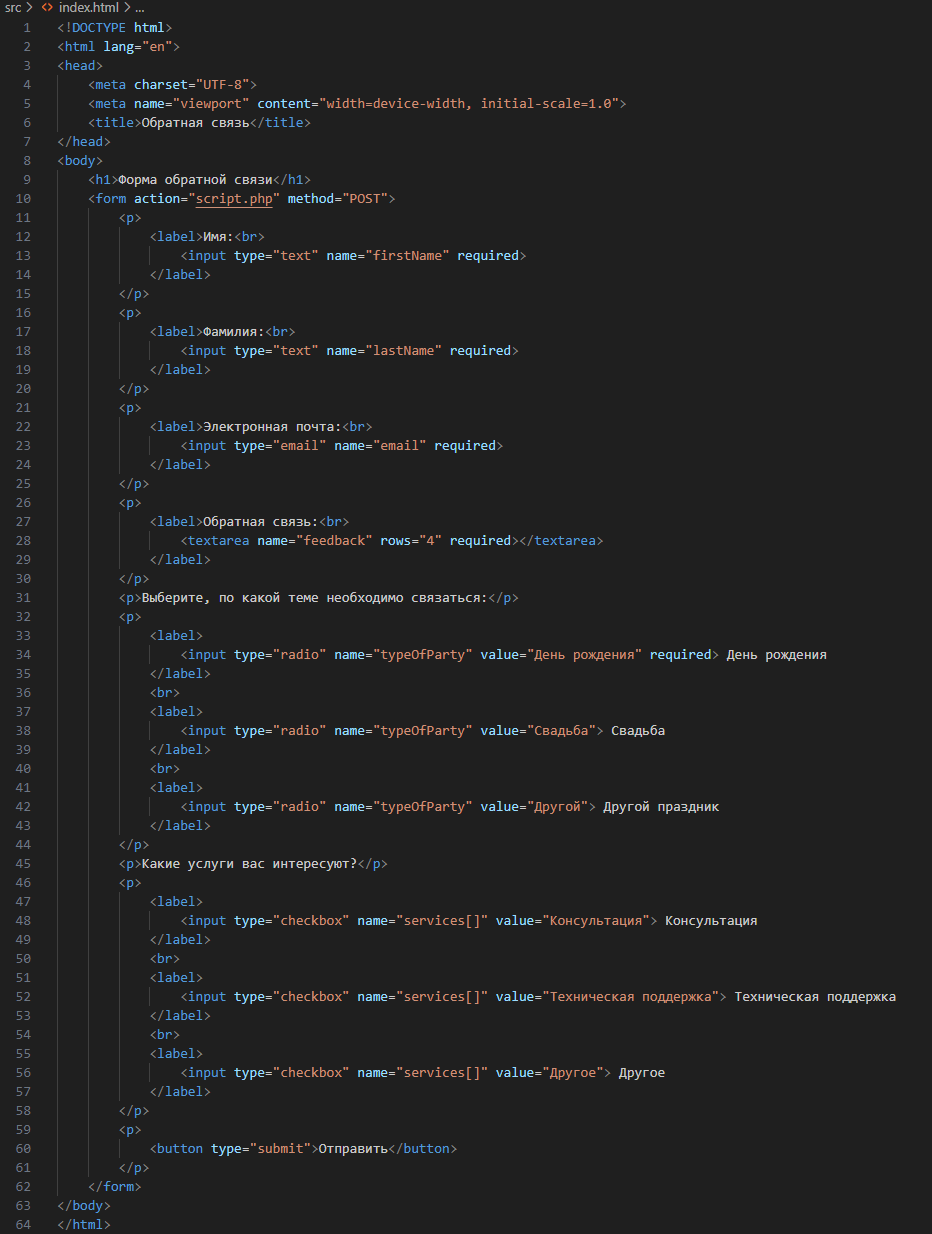


Рисунок 6 – HTML-код

Далее был написан php-скрипт (Рисунок 7). В post-запросе сначала проверяется что запрос является POST, все поля из формы записываются в переменные, поле с множественным выбором обрабатывается как массив, если оно пустое, возвращается пустой массив, данные форматируются и записываются в файл feedback.txt, после записи происходит перенаправление на страницу благодарности. В скрипте 2 get-запроса: оба зависят от переданного параметра и выводят либо страницу с благодарностью и кнопкой с отображением всех заявок (Рисунок 9), либо же страницу, которая выводит содержимое файла feedback.txt с сохранением форматирования (Рисунок 10).



Рисунок 7 – PHP-скрипт

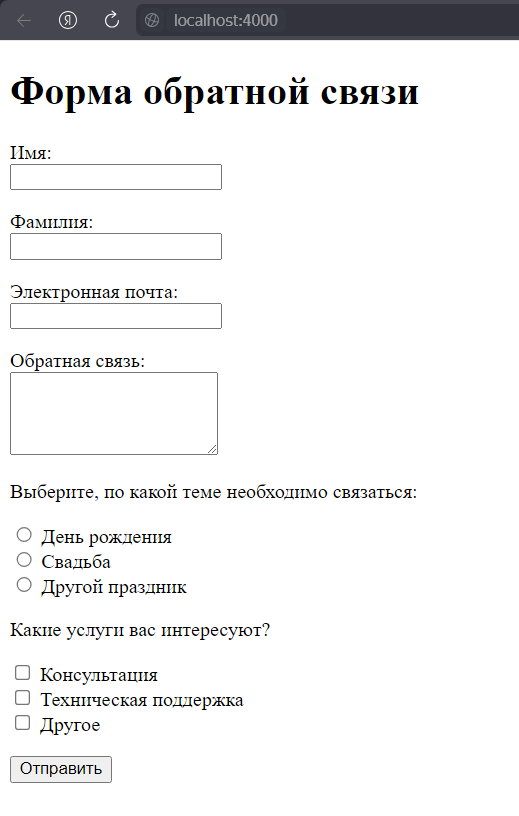


Рисунок 8 – Начальная страница

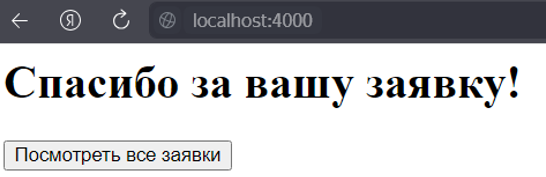


Рисунок 9 – Страница с благодарностью

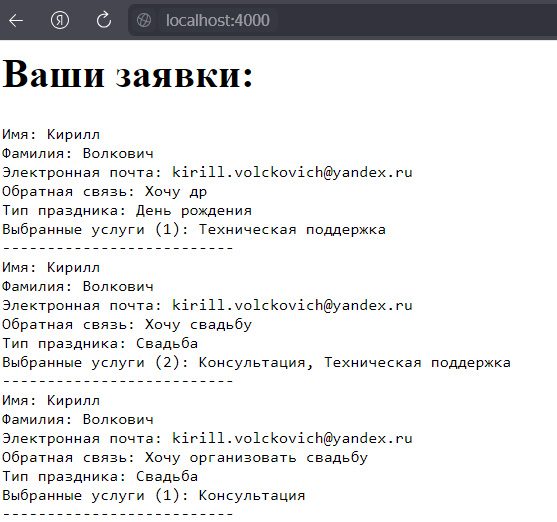


Рисунок 10 – Страница со всеми заявками

# Часть 3. WordPress

Для начала был скачен XMAPP, чтобы запустить локальный сервер. Далее был скачен движок WordPress и перенесен в папку htdocs. После всех установок необходимо было перейти по <http://localhost/phpmyadmin> и создать базу данных для сайта. После успешного создания БД был осуществлен переход по ссылке <http://localhost/wordpress>, где появлялась форма, в которой необходимо было указать имя созданной БД, имя пользователя и пароль. Далее в следующем окне необходимо было ввести всю необходимую информацию о сайте (Рисунок 11).

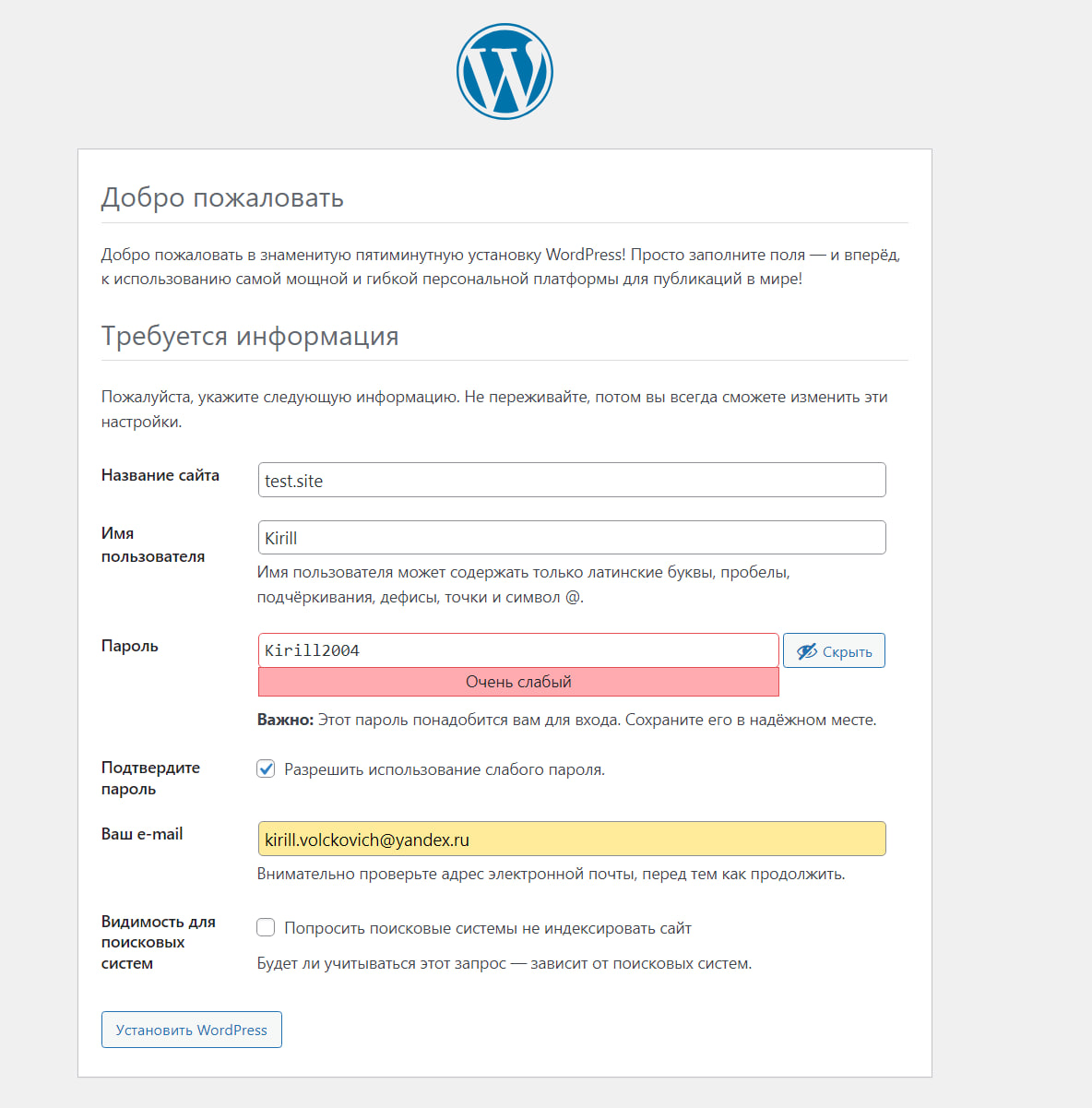


Рисунок 11 – WordPress

После указания всей необходимой информации открывается консоль разработчика для созданного сайта (Рисунок 12).

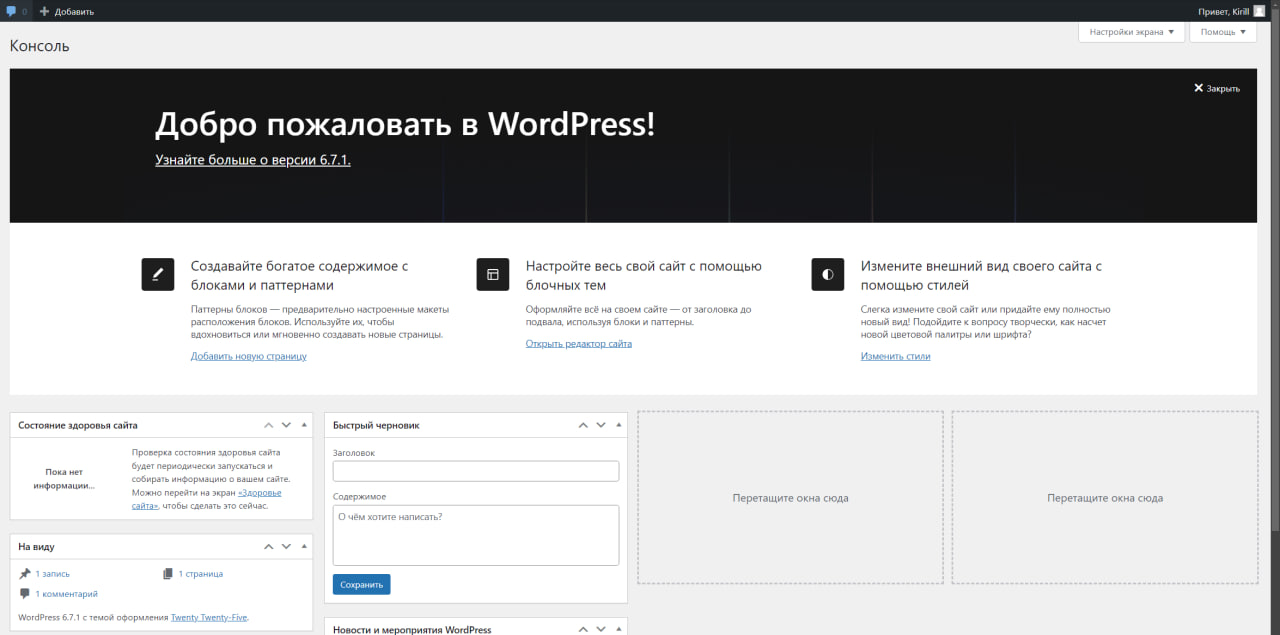


Рисунок 12 – Консоль разработчика

Далее были прописаны виртуальные хосты. В файле httpd-vhosts.conf было прописано два хоста (Рисунок 13).

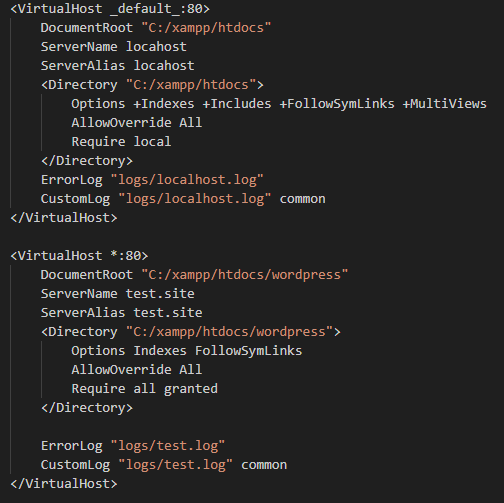


Рисунок 13 – Виртуальные хосты

Следующим шагом было изменение на листе wp\_options созданной БД двух переменных siteurl и home на <http://test.site>

После всех этих действий переходим по <http://test.site> и видим страницу сайта (Рисунок 14).

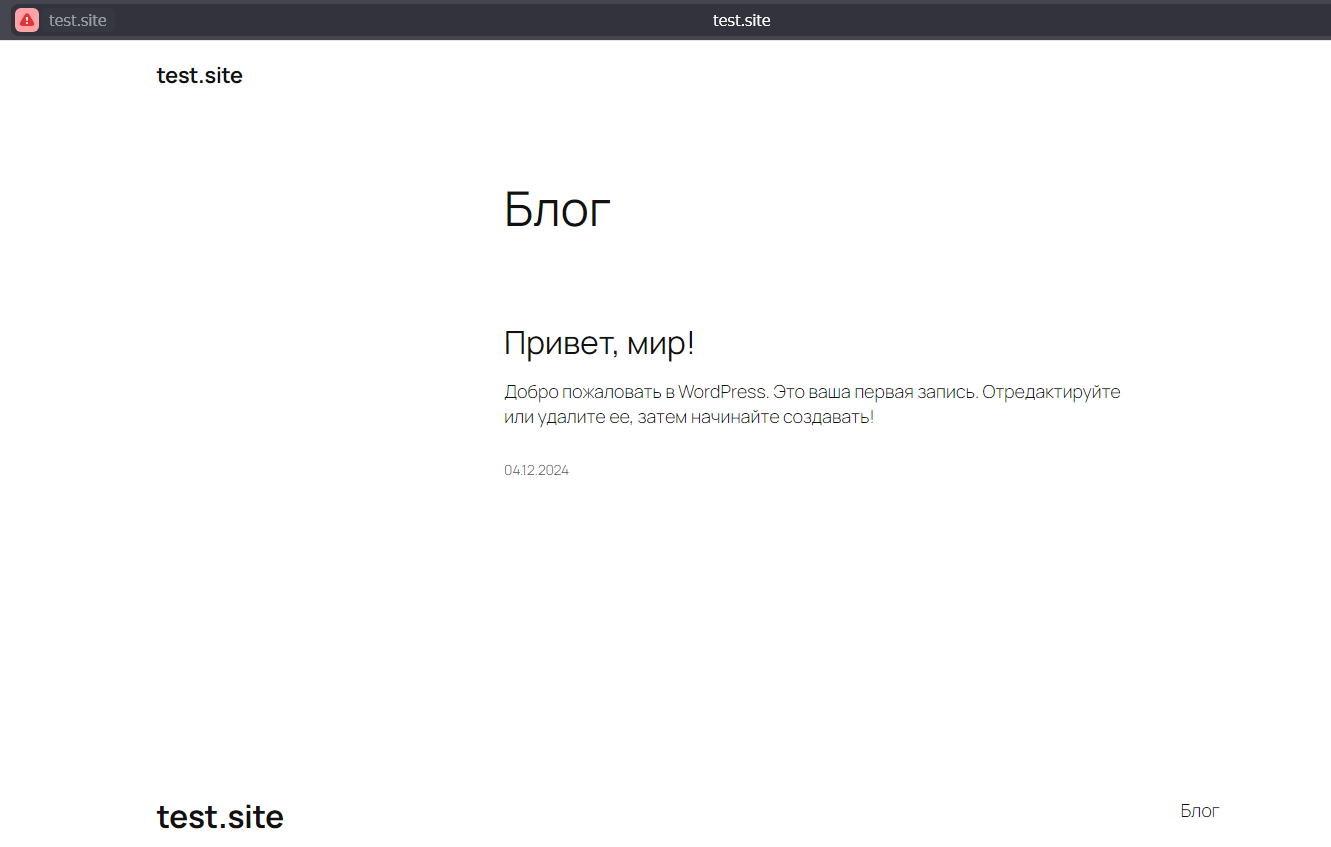


Рисунок 14 – Страница сайта

# Заключение

В результате выполнения лабораторной работы были изучены последовательные и параллельные задачи Gulp, был написан сайт для отправки формы обратной связи, изучены post и get запросы, а также ознакомление с XAMPP и WordPress и дальнейшая настройка перехода на сайт http://test.site.