

Санкт-Петербургский Национальный Исследовательский Университет

Информационных Технологий, Механики и Оптики

Факультет инфокоммуникационных технологий

**Лабораторная работа №3**

Выполнила: Ульянова Е.А.

Проверила: Марченко Е.В.

Санкт-Петербург,

2024

## Оглавление

Цель работы .....	3
Ход работы.....	3
1. Создание gulp-задач.....	3
2. Разработка html файла и php-скрипта.....	5
3. Использование движка wordpress для настройки портала .....	9
Заключение .....	13

## **Цель работы**

Целью данной лабораторной работы является получение новых знаний по работе с gulp, написанию скриптов на языке PHP и взаимодействию с инструментарием для отладки проектов и движком wordpress.

## **Ход работы**

### **1. Создание gulp-задач**

Для выполнения первого задания требуется создать два задачи и настроить их на последовательное и параллельное выполнение, а также настроить отображение файлов проекта в браузере и автоматическую перезагрузку при изменении файлов проекта.

Для реализации этого был создан gulpfile, в котором использовались такие функции пакета Gulp, как series(), parallel() и watch(). Также были созданы файлы index.html и style.css, находящиеся в папке src и необходимые для проверки работоспособности последней из вышеописанных функции (при их изменении с помощью watch() вызывается метод browserSync.reload, который обновляет страницу в браузере). Код gulpfile представлен на рисунке 1.

```

1  const gulp = require('gulp');
2  const browserSync = require('browser-sync').create();
3
4  function task1(done) {
5    console.log('a');
6    done();
7  }
8  function task2(done) {
9    console.log('b');
10   done();
11 }
12
13 function watchFiles() {
14   browserSync.init({
15     server: {
16       baseDir: "./src"
17     }
18   });
19
20   gulp.watch("./src/*.html").on('change', browserSync.reload);
21   gulp.watch("./src/*.css").on('change', browserSync.reload);
22 }
23
24 exports.series = gulp.series(task1, task2);
25 exports.parallel = gulp.parallel(task1, task2);
26 exports.watch = watchFiles;

```

Рисунок 1 - gulpfile.js

Результат запуска `series` и `parallel` из командной строки представлен на рисунке 2. Ожидаемо в случае параллельного выполнения Gulp показывает одновременный запуск двух задач, а при использовании функции `series()` выполнение новой задачи стартует только после окончания выполнения предыдущей.

```

C:\Users\Ekaterina>cd "C:\Users\Ekaterina\Desktop\Учебные задания\3 курс\Web\Lab3\task_1"
C:\Users\Ekaterina\Desktop\Учебные задания\3 курс\Web\Lab3\task_1>gulp parallel
[14:53:09] Using gulpfile ~\Desktop\Учебные задания\3 курс\Web\Lab3\task_1\gulpfile.js
[14:53:09] Starting 'parallel'...
[14:53:09] Starting 'task1'...
[14:53:09] Starting 'task2'...
a
[14:53:09] Finished 'task1' after 3.02 ms
b
[14:53:09] Finished 'task2' after 3.67 ms
[14:53:09] Finished 'parallel' after 9.63 ms
C:\Users\Ekaterina\Desktop\Учебные задания\3 курс\Web\Lab3\task_1>gulp series
[14:53:20] Using gulpfile ~\Desktop\Учебные задания\3 курс\Web\Lab3\task_1\gulpfile.js
[14:53:20] Starting 'series'...
[14:53:20] Starting 'task1'...
a
[14:53:20] Finished 'task1' after 2.8 ms
[14:53:20] Starting 'task2'...
b
[14:53:20] Finished 'task2' after 1.57 ms
[14:53:20] Finished 'series' after 12 ms

```

Рисунок 2 - Результат последовательного и параллельного выполнения задач

Результат запуска watch представлен на рисунке 3.

```
C:\Users\Ekaterina\Desktop\Учебные задания\3 курс\Web\Lab3\task_1>gulp watch
[15:01:47] Using gulpfile ~\Desktop\Учебные задания\3 курс\Web\Lab3\task_1\gulpfile.js
[15:01:47] Starting 'watch'...
[Browsersync] Access URLs:
=====
  Local: http://localhost:3000
  External: http://192.168.56.1:3000
=====
   UI: http://localhost:3001
  UI External: http://192.168.56.1:3001
=====
[Browsersync] Serving files from: ./src
[Browsersync] File event [change] : src\style.css
[Browsersync] Reloading Browsers...
[Browsersync] Reloading Browsers...
```

Рисунок 3 – Запуск watch и внесение некоторых изменений в файлы

## 2. Разработка html файла и php-скрипта

Для выполнения второго задания требуется создать форму для отправки информации по обратной связи от пользователя сайта.

Код html файла, содержащего в себе форму, использующую метод POST для отправки данных, представлен на рисунке 4 (написан по образцу файла ex7.html, что был представлен в первой лабораторной работе, однако дополнительно содержит функцию validateForm(), которая предотвращает отправку формы, в которой ни один из вариантов checkbox не выбран).



Рисунок 4 - Файл form.html

Однако важно отметить, что данные формы могли быть отправлены и с использованием метода GET (пример с ним также будет приведён в отчёте), поэтому следует учитывать некоторые их особенности:

- GET: параметры передаются в URL. Ограничение по размеру запроса зависит от браузера и сервера. Данные видны в URL, что делает их менее безопасными.
- POST: информация передается в теле запроса. POST-запросы могут передавать большие объемы данных без ограничений, чем GET-запросы. Данные не видны в URL, что делает их более безопасными для передачи чувствительной информации.

Также был написан php-скрипт для обработки post-запроса, в котором происходит получение данных формы и вывод персонализированного сообщения благодарности за её заполнение. Результат представлен на рисунке 5.

```

1  <?php
2  if ($_SERVER['REQUEST_METHOD'] === 'POST') {
3      $first_name = htmlspecialchars($_POST['first_name']);
4      $last_name = htmlspecialchars($_POST['last_name']);
5      $email = htmlspecialchars($_POST['email']);
6      $feedback = htmlspecialchars($_POST['feedback']);
7      $category = htmlspecialchars($_POST['category']);
8      $topics = isset($_POST['topics']) ? $_POST['topics'] : [];
9
10     echo "<h1>Спасибо за ваш отзыв, $first_name</h1>";
11 } else {
12     echo "<h1>Ошибка: Данные не отправлены.</h1>";
13 }
14 ?>
15

```

Рисунок 5 - php-скрипт

Тестирование производилось с использованием MAMP (Macintosh, Apache, MySQL, PHP) - бесплатной локальной серверной среды, которую можно установить под macOS и Windows. Результаты представлены на рисунках 6-7.

← → ↻ localhost/feedback/form.html

**Заполните форму для отправки отзыва по выбранному товару:**

Имя:

Фамилия:

Электронная почта:

Обратная связь:

Выберите категорию товара:

☒ Бумажный

☐ Электронный

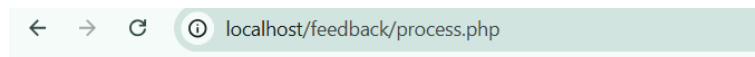
Выберите наименования из набора:

☐ Собачье сердце

☒ Мёртвые души

☒ Портрет Дориана Грея

Рисунок 6 - Форма обратной связи



## Спасибо за ваш отзыв, Екатерина

Рисунок 7 - Результат отправки заполненной формы

Для демонстрации сравнения методов GET и POST, описанных ранее, были созданы два файла: form.html и process.php, расположенные в папке task2\_with\_GET. Код представлен на рисунках 8-9.

```
index.html x form.html x form.html x
1 <!DOCTYPE html>
2 <html lang="ru">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Форма GET</title>
7 </head>
8 <body>
9   <h1>Отправка данных через GET</h1>
10  <form action="process.php" method="GET">
11    <label for="name">Имя:</label>
12    <input type="text" id="name" name="name" required>
13    <br><br>
14    <label for="email">Email:</label>
15    <input type="email" id="email" name="email" required>
16    <br><br>
17    <input type="submit" value="Отправить">
18  </form>
19 </body>
20 </html>
```

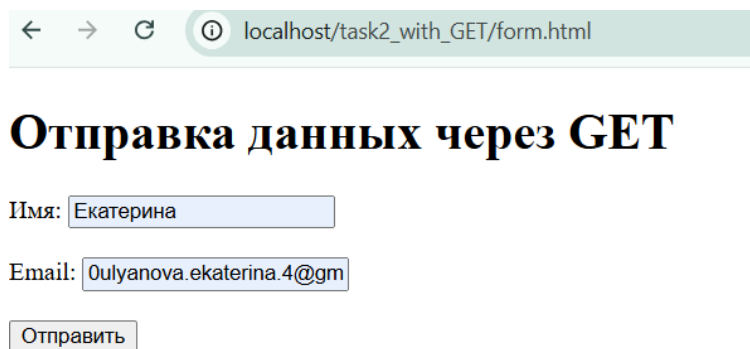
Рисунок 8 - Файл form.html (с методом GET)

```
1 <?php
2 if ($_SERVER["REQUEST_METHOD"] == "GET") {
3     $name = htmlspecialchars($_GET['name']);
4     $email = htmlspecialchars($_GET['email']);
5
6     echo "<h1>Полученные данные</h1>";
7     echo "<p><strong>Имя:</strong> " . $name . "</p>";
8     echo "<p><strong>Email:</strong> " . $email . "</p>";
9 } else {
10    echo "Данные не были отправлены.";
11 }
12 ?>
```

Рисунок 9 - Файл process.php (с методом GET)

Тестирование проводилось аналогичным способом. Результаты представлены на рисунках 10-11.





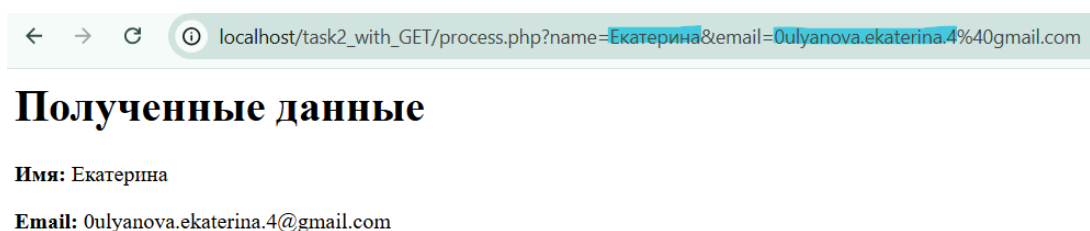
← → ↻ ⓘ localhost/task2\_with\_GET/form.html

## Отправка данных через GET

Имя:

Email:

Рисунок 10 - Новая форма



← → ↻ ⓘ localhost/task2\_with\_GET/process.php?name=Екатерина&email=Oulyanova.ekaterina.4%40gmail.com

## Полученные данные

**Имя:** Екатерина

**Email:** Oulyanova.ekaterina.4@gmail.com

Рисунок 11 - Результат отправки формы (с методом GET)

Действительно, при использовании метода GET параметры передаются в URL, чего не происходит при выборе POST.

### 3. Использование движка wordpress для настройки портала

Для выполнения третьего задания необходимо установить инструментарий для отладки проектов. Было принято решение воспользоваться МАМР, который ранее уже был задействован во втором задании. Также было произведено скачивание движка wordpress. Для его дальнейшего запуска папка с ним была помещена в папку htdocs. В последствии был осуществлён переход на <http://localhost/phpMyAdmin5/> для создания базы данных, необходимой для настройки wordpress. Наличие базы отображено на рисунке 12.

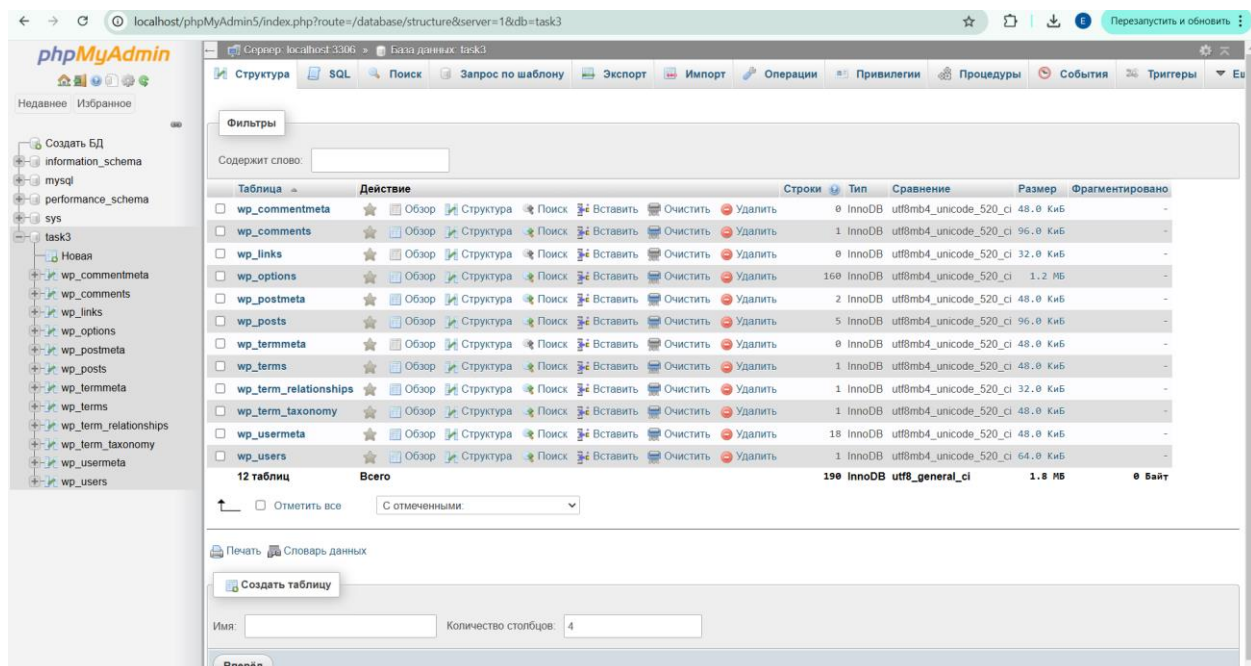


Рисунок 12 - Созданная база данных

Затем по адресу <http://localhost/wordpress> была заполнена появившаяся форма. Она представлена на рисунке 13.

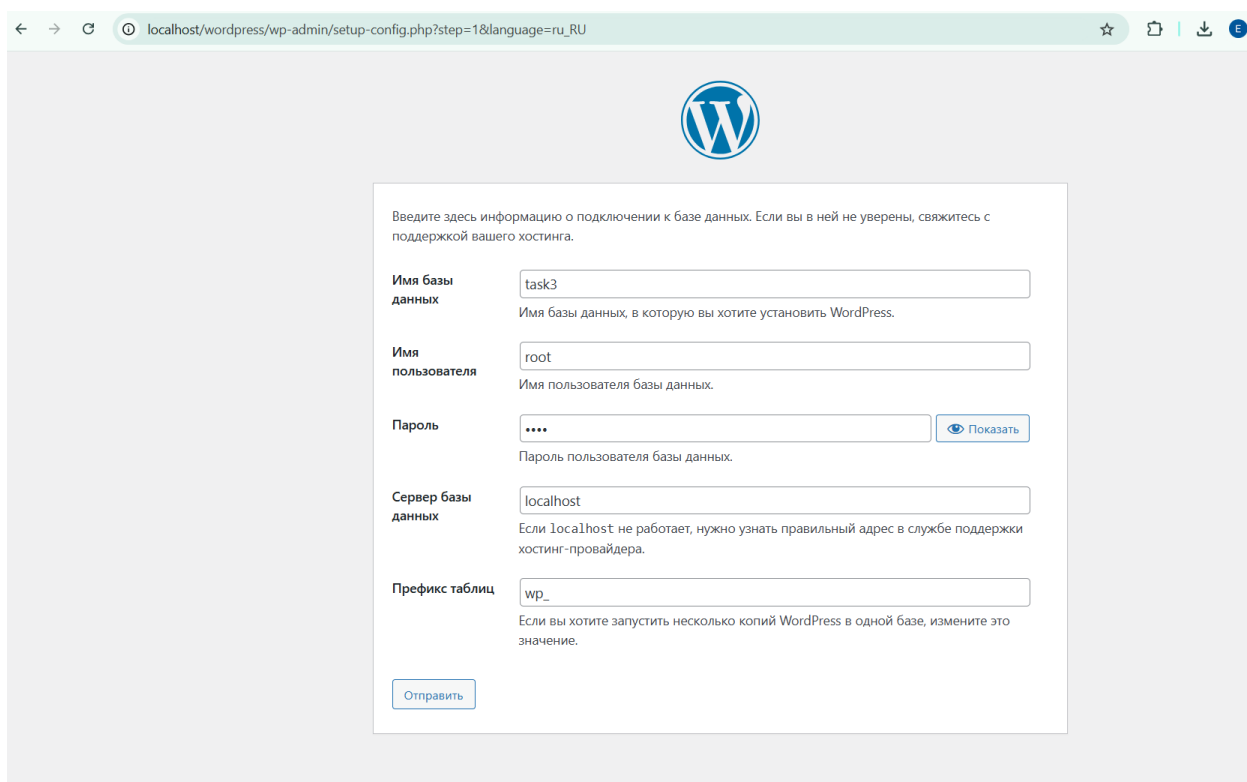


Рисунок 13 - Заполнение формы

Далее была заполнена информация о создаваемом сайте (рисунок 14).

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.

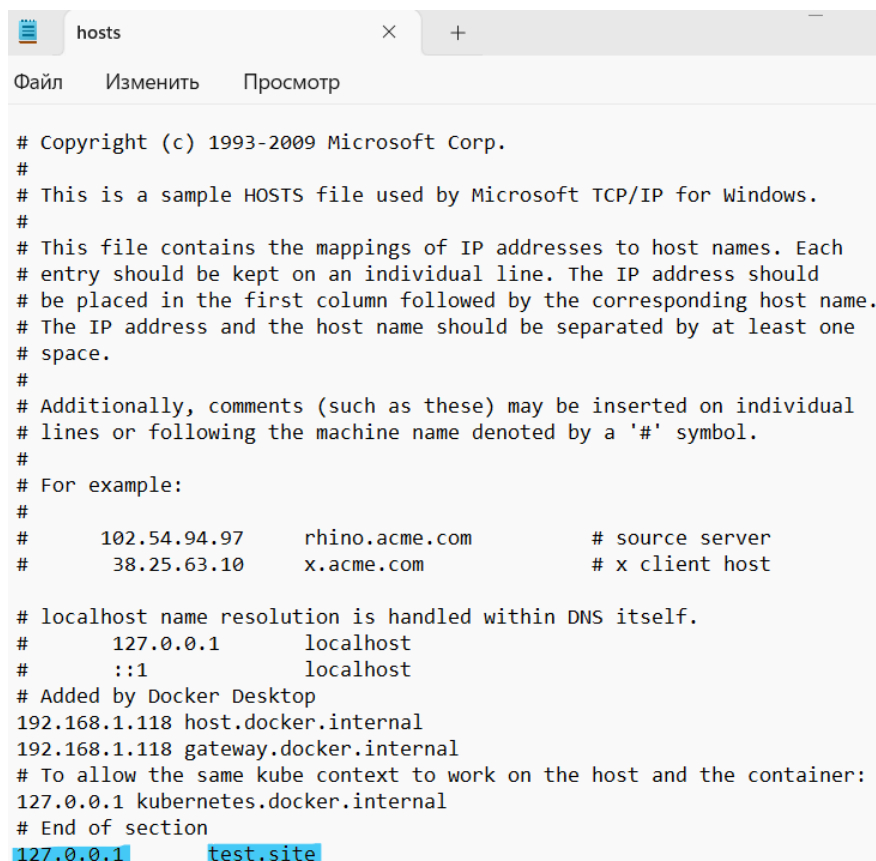
[Установить WordPress](#)

Рисунок 14 - Внесение информации о создаваемом сайте

На рисунках 15-18 приведены действия, которые также необходимо выполнить для того, чтобы создаваемый сайт был доступен по ссылке <http://test.site> (настройка виртуальных хостов, редактирование файла hosts, добавление конфигурационного файла и изменение адреса сайта).

```
40 <VirtualHost *:80>
41 |     ServerName test.site
42 |     DocumentRoot "C:/MAMP/htdocs/wordpress"
43 </VirtualHost>
44
45 <VirtualHost *:80>
46 |     ServerName localhost
47 |     DocumentRoot "C:/MAMP/htdocs"
48 </VirtualHost>
```

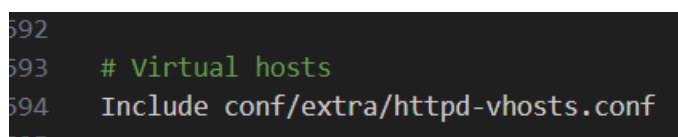
Рисунок 15 - Настройка виртуальных хостов



```
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#       102.54.94.97       rhino.acme.com           # source server
#       38.25.63.10       x.acme.com               # x client host

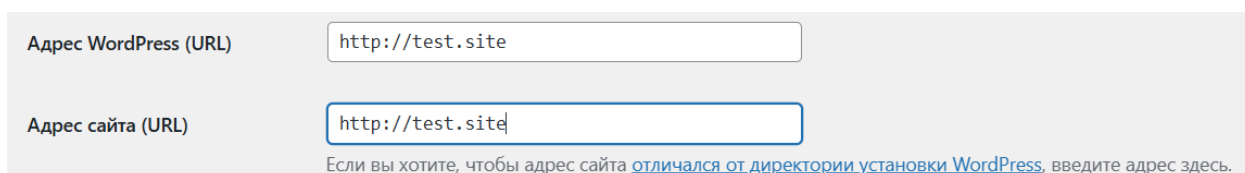
# localhost name resolution is handled within DNS itself.
#       127.0.0.1        localhost
#       ::1              localhost
# Added by Docker Desktop
192.168.1.118 host.docker.internal
192.168.1.118 gateway.docker.internal
# To allow the same kube context to work on the host and the container:
127.0.0.1 kubernetes.docker.internal
# End of section
127.0.0.1 test.site
```

Рисунок 16 – Дополнение файла hosts



```
592
593 # Virtual hosts
594 Include conf/extra/httpd-vhosts.conf
595
```

Рисунок 17 - Добавление конфигурационного файла (по умолчанию был отключён)



Адрес WordPress (URL)

Адрес сайта (URL)

Если вы хотите, чтобы адрес сайта [отличался от директории установки WordPress](#), введите адрес здесь.

Рисунок 18 - Изменение адреса сайта

## **Заключение**

В ходе данной лабораторной работы были получены новые знания по работе с gulp, а именно по использованию встроенных функций series(), parallel() и watch(); написанию скриптов на языке PHP для обработки post- и get-запросов и взаимодействию с инструментарием для тестирования и отладки проектов MAMP и движком wordpress.