

Санкт-Петербургский Национальный Исследовательский Университет
Информационных Технологий, Механики и Оптики

Факультет Инфокоммуникационных Технологий

Лабораторная работа №3

Выполнил:

Теребов М.А.

Проверила:

Марченко Е.В.

Санкт-Петербург, 2024

Оглавление

Цель Работы	3
Ход работы	4
Заключение	17

Цель Работы

Изучить инструменты gulp, wordpress, познакомиться с php, ХАМРР.

Ход работы

1. Работа с Gulp

Для начала был создан проект имеющий структуру представленную на рисунке ниже.

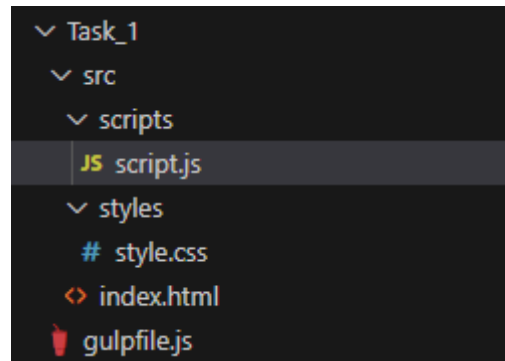


Рисунок 1 – структура проекта

Далее был инициализирован gulp и установлен browser-sync, помогающий отслеживать изменения в файлах и автоматически отображать их в браузере.

```
PS C:\Users\Max\Documents\ITMO\web\Lab_3\Task_1> npm init -y
Wrote to C:\Users\Max\Documents\ITMO\web\Lab_3\Task_1\package.json:

{
  "name": "task_1",
  "version": "1.0.0",
  "main": "gulpfile.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}

PS C:\Users\Max\Documents\ITMO\web\Lab_3\Task_1> npm install --save-dev gulp gulp-sass browser-sync gulp-concat
added 290 packages, and audited 291 packages in 9s

15 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\Max\Documents\ITMO\web\Lab_3\Task_1>
```

Рисунок 2 – инициализация и установка gulp

Были добавлены task'и в gulpfile.js для работы с CSS, HTML, JS.

```

// Задача для обработки HTML
function htmlTask() {
  console.log("Task_1")
  return gulp.src(paths.html)
    .pipe(gulp.dest('./dist'))
    .pipe(browserSync.stream());
}

// Задача для обработки CSS
function cssTask() {
  console.log("Task_2")
  return gulp.src(paths.css)
    .pipe(concat('style.min.css'))
    .pipe(gulp.dest('./dist/styles'))
    .pipe(browserSync.stream());
}

// Задача для обработки JS
function jsTask() {
  console.log("Task_3")
  return gulp.src(paths.js)
    .pipe(concat('script.min.js'))
    .pipe(gulp.dest('./dist/scripts'))
    .pipe(browserSync.stream());
}

```

Рисунок 3 – Добавление задач

Протестируем последовательное и параллельное выполнение задач, для последовательного используем `gulp.series`, а для параллельного – `gulp.parallel`. Разница и результаты выполнения на рисунках ниже.

```

PS C:\Users\Max\Documents\ITMO\web\Lab_3\Task_1> npx gulp build
[ ] Using gulpfile
[ ] Starting 'build'...
[ ] Starting 'htmlTask'...
Task_1
[ ] Finished 'htmlTask' after
[ ] Starting 'cssTask'...
Task_2
(node:11656) [DEP0180] DeprecationWarning: fs.Stats constructor is deprecated.
(Use `node --trace-deprecation ...` to show where the warning was created)
[ ] Finished 'cssTask' after
[ ] Starting 'jsTask'...
Task_3
[ ] Finished 'jsTask' after
[ ] Finished 'build' after

```

Рисунок 4 – `gulp.series`

```

PS C:\Users\Max\Documents\ITMO\web\Lab_3\Task_1> npx gulp dev
    [ ] Using gulpfile
    [ ] Starting 'dev'...
    [ ] Starting 'htmlTask'...
    [ ] Starting 'cssTask'...
    [ ] Starting 'jsTask'...
    [ ] Starting 'serve'...
task_1
task_2
task_3
(node:18284) [DEP0180] DeprecationWarning: fs.Stats constructor is deprecated.
(Use `node --trace-deprecation ...` to show where the warning was created)
[Browsersync] Access URLs:
-----
    Local:
    External:
-----
    UI:
    UI External:
-----
[Browsersync] Serving files from:
[Browsersync] 1 file changed (
    [ ] Finished 'htmlTask' after
[Browsersync] 1 file changed (
    [ ] Finished 'cssTask' after
[Browsersync] 1 file changed (
    [ ] Finished 'jsTask' after
[Browsersync] Reloading Browsers... (buffered 3 events)

```

Рисунок 5 – *gulp.parallel*

Также был ревизован таск автоматической перезагрузки при изменении одного из контролируемых файлов проекта, код таска представлен ниже.

```

// Таск для запуска BrowserSync
function serve() {
  browserSync.init({
    server: {
      baseDir: './dist',
    },
  });

  // Наблюдение за файлами
  gulp.watch(paths.html, htmlTask);
  gulp.watch(paths.css, cssTask);
  gulp.watch(paths.js, jsTask);
}

```

Рисунок 6 – Таск для автоматической перезагрузки

Произведем проверку – после изменения css файла, цвет заднего фона поменялся на странице автоматически.

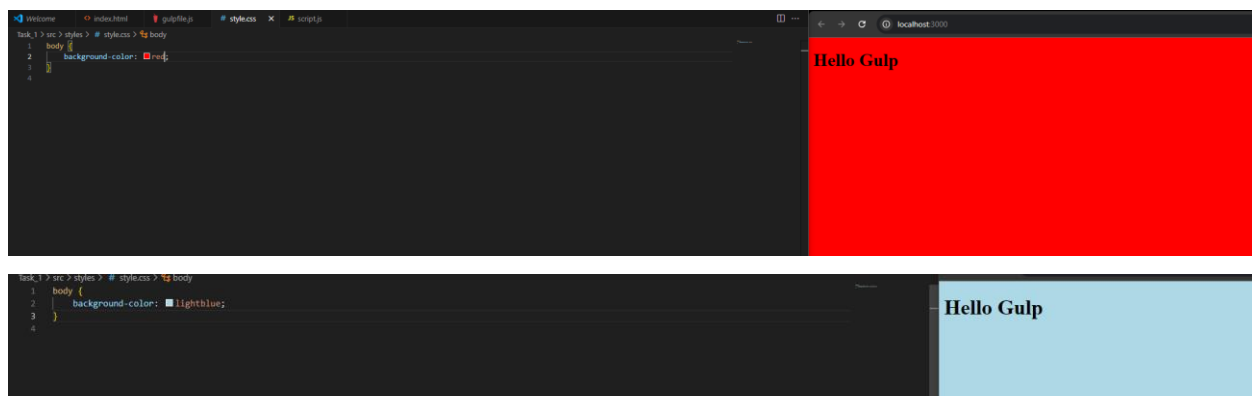


Рисунок 7 – Проверка автоматической перезагрузки

2. Создание формы

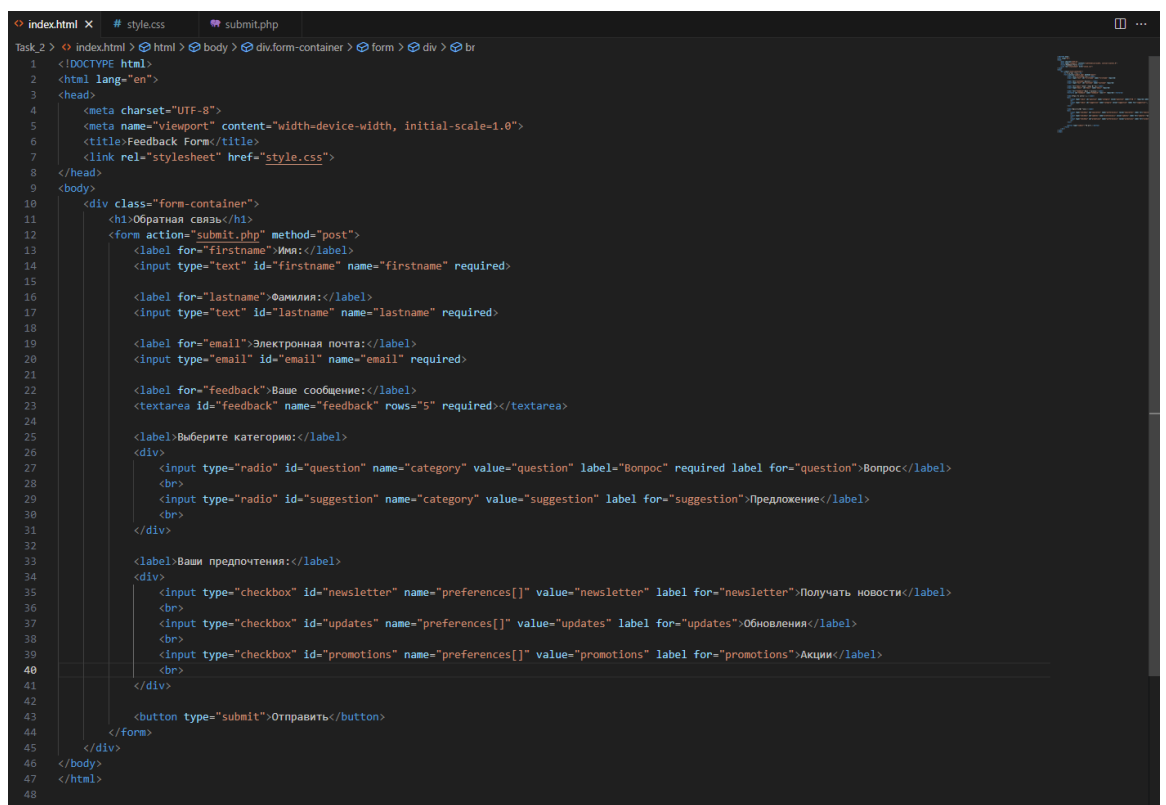
Для выполнения этого упражнения был установлен php и добавлен в path windows. Проверим правильность установки.

```
C:\Users\Max\Documents\ITMO\web\Lab_3\Task_2>php -v
PHP 8.4.2 (cli) (built: Dec 17 2024 17:28:28) (NTS Visual C++ 2022 x64)
Copyright (c) The PHP Group
Zend Engine v4.4.2, Copyright (c) Zend Technologies

C:\Users\Max\Documents\ITMO\web\Lab_3\Task_2>_
```

Рисунок 8 – проверка установки php

Далее создаем проект с php скриптом css файлом стилей и html страницей. На рисунках ниже представлены программные коды, содержащиеся в них.



```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <meta name="viewport" content="width=device-width, initial-scale=1.0">
6 <title>Feedback Form</title>
7 <link rel="stylesheet" href="style.css">
8 </head>
9 <body>
10 <div class="form-container">
11 <h1>Обратная связь</h1>
12 <form action="submit.php" method="post">
13 <label for="firstname">Имя:</label>
14 <input type="text" id="firstname" name="firstname" required>
15
16 <label for="lastname">Фамилия:</label>
17 <input type="text" id="lastname" name="lastname" required>
18
19 <label for="email">Электронная почта:</label>
20 <input type="email" id="email" name="email" required>
21
22 <label for="feedback">Ваше сообщение:</label>
23 <textarea id="feedback" name="feedback" rows="5" required></textarea>
24
25 <label>Выберите категорию:</label>
26 <div>
27 <input type="radio" id="question" name="category" value="question" label="Вопрос" required label for="question">Вопрос</label>
28 <br>
29 <input type="radio" id="suggestion" name="category" value="suggestion" label for="suggestion">Предложение</label>
30 <br>
31 </div>
32
33 <label>Ваши предпочтения:</label>
34 <div>
35 <input type="checkbox" id="newsletter" name="preferences[]" value="newsletter" label for="newsletter">Получать новости</label>
36 <br>
37 <input type="checkbox" id="updates" name="preferences[]" value="updates" label for="updates">Обновления</label>
38 <br>
39 <input type="checkbox" id="promotions" name="preferences[]" value="promotions" label for="promotions">Акции</label>
40 <br>
41 </div>
42 <button type="submit">Отправить</button>
43 </form>
44 </div>
45 </body>
46 </html>
```

Рисунок 9 – Программный код index.html


```

Task_2 > # style.css > ...
1 body {
2   font-family: Arial, sans-serif;
3   background-color: #f9f9f9;
4   margin: 0;
5   padding: 0;
6 }
7
8 .form-container {
9   max-width: 500px;
10  margin: 50px auto;
11  background: #ffffff;
12  padding: 20px;
13  border-radius: 8px;
14  box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
15 }
16
17 h1 {
18   text-align: center;
19   color: #333333;
20 }
21
22 label {
23   display: block;
24   margin-top: 15px;
25   font-weight: bold;
26 }
27
28 input[type="text"],
29 input[type="email"],
30 textarea {
31   width: 100%;
32   padding: 10px;
33   margin-top: 5px;
34   border: 1px solid #cccccc;
35   border-radius: 4px;
36 }
37
38 button {
39   width: 100%;
40   padding: 10px;
41   margin-top: 20px;
42   background-color: #007bff;
43   color: #ffffff;
44   border: none;
45   border-radius: 4px;
46   cursor: pointer;
47   font-size: 16px;
48 }
49
50 button:hover {
51   background-color: #0056b3;
52 }
53
54 input[type="radio"],
55 input[type="checkbox"] {
56   margin-right: 10px;
57 }
58

```

Рисунок 10 – Программный код style.css

```

Task_2 > submit.php
1 <?php
2 if ($_SERVER["REQUEST_METHOD"] === "POST") {
3   // Получение данных
4   $firstname = htmlspecialchars($_POST['firstname']);
5   $lastname = htmlspecialchars($_POST['lastname']);
6   $email = htmlspecialchars($_POST['email']);
7   $feedback = htmlspecialchars($_POST['feedback']);
8   $category = htmlspecialchars($_POST['category']);
9   $preferences = isset($_POST['preferences']) ? $_POST['preferences'] : [];
10
11   echo "<h1>Ваши данные (POST):</h1>";
12   echo "Имя: $firstname<br>";
13   echo "Фамилия: $lastname<br>";
14   echo "Электронная почта: $email<br>";
15   echo "Сообщение: $feedback<br>";
16   echo "Категория: $category<br>";
17   echo "Предпочтения: " . implode(", ", $preferences) . "<br>";
18 } elseif ($_SERVER["REQUEST_METHOD"] === "GET") {
19   echo "<h1>Вы использовали GET</h1>";
20 } else {
21   echo "<h1>Неизвестный метод</h1>";
22 }
23 ?>
24

```

Рисунок 11 – Программный код submit.php

Проверим работу страницы. Для начала запускаем локальный сервер с помощью “php -S localhost:8000”, и переходим по адресу <http://localhost:8000/index.html>. Результат на рисунке ниже.

Обратная связь

Имя:

MAKSIM

Фамилия:

MAKSIM

Электронная почта:

mterebov@gmail.com

Ваше сообщение:

asfd

Выберите категорию:

☒ Вопрос

☐ Предложение

Ваши предпочтения:

☐ Получать новости

☒ Обновления

☒ Акции

Отправить

Рисунок 12 – Веб-страница

GET

([http://localhost:8000/submit.php?firstname=Иван&lastname=Иванов&email=ivan@example.com&feedback=Привет&category=question&preferences\[\]=newsletter&preferences\[\]=updates](http://localhost:8000/submit.php?firstname=Иван&lastname=Иванов&email=ivan@example.com&feedback=Привет&category=question&preferences[]=newsletter&preferences[]=updates)) и POST (просто заполним форму), они представлены на рисунках 13 и 14 соответственно.

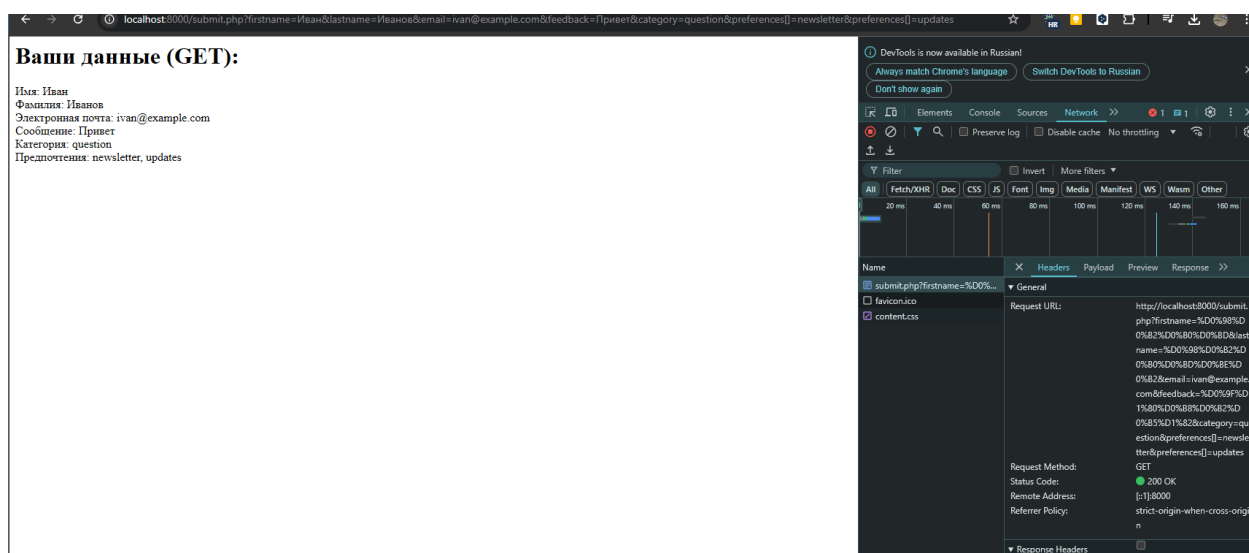


Рисунок 13 – метод GET

Ваши данные (POST):

Имя: MAKSIM
Фамилия: MAKSIM
Электронная почта: mterebov@gmail.com
Сообщение: asfd
Категория: question
Предпочтения: updates, promotions

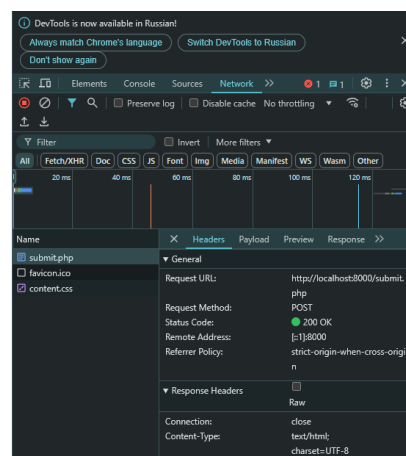


Рисунок 14 – метод POST

Методы GET и POST используются в HTTP-протоколе для отправки данных с клиента на сервер. Оба метода являются основными средствами для взаимодействия между веб-страницами и сервером при отправке данных.

Метод GET отправляет данные в URL-строке запроса (в адресной строке браузера). Этот метод используется для получения информации с сервера, например, для запросов на поиск или при переходе по страницам.

Особенности метода GET:

- Данные в URL: Все данные передаются как часть URL в строке запроса (например: `https://example.com?name=John&email=john@example.com`).
- Ограничения на размер данных: Поскольку данные передаются через URL, существует ограничение на размер данных (обычно около 2000 символов, но это зависит от браузера и сервера).
- Безопасность: Данные, передаваемые через GET, видны в адресной строке и могут быть перехвачены, что делает метод менее безопасным для передачи конфиденциальной информации.
- Кэширование: Запросы, использующие GET, могут быть кэшированы браузером или прокси-серверами, что позволяет ускорить повторные запросы.
- Идемпотентность: GET-запросы считаются идемпотентными, что означает, что повторный запрос с теми же параметрами не должен изменять состояние на сервере.

Метод POST отправляет данные в теле запроса, что позволяет передавать более сложные и объемные данные. Этот метод чаще всего используется для отправки данных, например, при заполнении формы на сайте.

Особенности метода POST:

- Данные в теле запроса: Данные не отображаются в адресной строке браузера, что делает метод более безопасным для передачи конфиденциальной информации.
- Отсутствие ограничений по размеру данных: В отличие от GET, в POST-запросе нет таких строгих ограничений на размер передаваемых данных.

- Безопасность: Данные, передаваемые через POST, не видны в URL и передаются в теле запроса, что делает метод более безопасным для отправки чувствительной информации (например, паролей).
- Не кэшируется: Запросы POST не кэшируются браузером.
- Не идемпотентность: Запросы POST могут изменять состояние на сервере, например, добавлять новые записи в базу данных, поэтому их повторное выполнение может привести к различным результатам.

3. Установка движка

Для выполнения задания был установлен локальный сервер **XAMPP**.

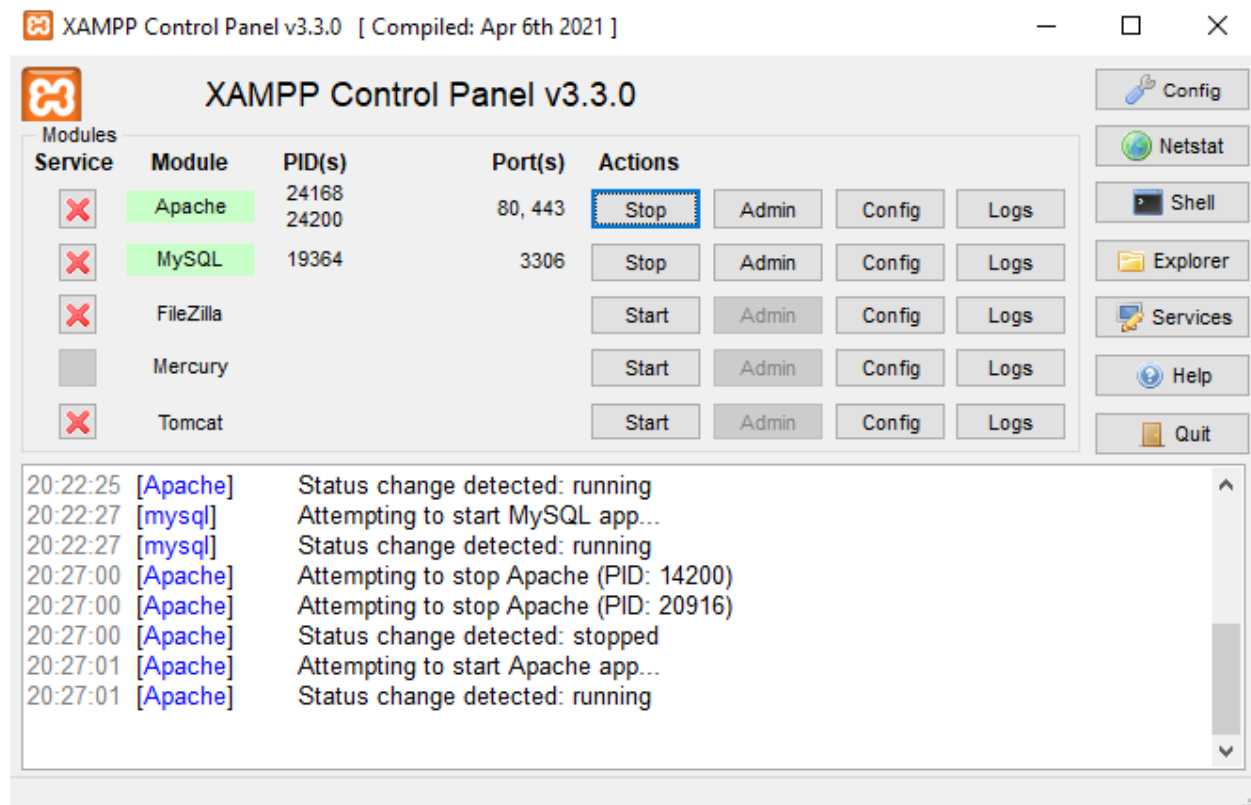


Рисунок 15 – XAMPP

Далее был создан локальный домен test.site. Был открыт файл hosts (C:\Windows\System32\drivers\etc\hosts) и в него добавлена строка “127.0.0.1 test.site”.

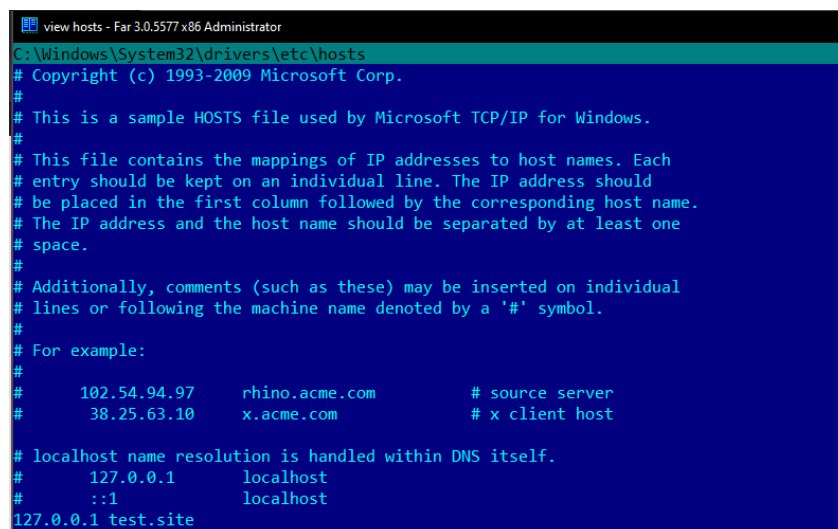


Рисунок 16 – hosts

Также виртуальный хостинг был добавлен в апач (C:\xampp\apache\conf\extra\httpd-vhosts.conf).

```
www.htpd-vhosts.conf - 12/10/2017 18:00:00
# Virtual Hosts
#
# Required modules: mod_log_config
#
# If you want to maintain multiple domains/hostnames on your
# machine you can setup VirtualHost containers for them. Most configurations
# use only name-based virtual hosts so the server doesn't need to worry about
# IP addresses. This is indicated by the asterisks in the directives below.
#
# Please see the documentation at
# <URL:http://httpd.apache.org/docs/2.4/vhosts/>
# for further details before you try to setup virtual hosts.
#
# You may use the command line option '-S' to verify your virtual host
# configuration.
#
# Use name-based virtual hosting.
#
#NameVirtualHost *:80
#
# VirtualHost example:
# Almost any Apache directive may go into a VirtualHost container.
# The first VirtualHost section is used for all requests that do not
# match a #ServerName or #ServerAlias in any <VirtualHost> block.
#
#<VirtualHost *:80>
#    #ServerAdmin webmaster@dummy-host.example.com
#    #DocumentRoot "/C:/xampp/htdocs/dummy-host.example.com"
#    #ServerName dummy-host.example.com
#    #ServerAlias www.dummy-host.example.com
#    #ErrorLog "logs/dummy-host.example.com-error.log"
#    #CustomLog "logs/dummy-host.example.com-access.log" common
#</VirtualHost>
#
#<VirtualHost *:80>
#    #ServerAdmin webmaster@dummy-host2.example.com
#    #DocumentRoot "/C:/xampp/htdocs/dummy-host2.example.com"
#    #ServerName dummy-host2.example.com
#    #ErrorLog "logs/dummy-host2.example.com-error.log"
#    #CustomLog "logs/dummy-host2.example.com-access.log" common
#</VirtualHost>
#
#<VirtualHost *:80>
#    DocumentRoot "/C:/xampp/htdocs/test_site"
#    ServerName test.site
#    <Directory "/C:/xampp/htdocs/test_site">
#        AllowOverride All
#        Require all granted
#    </Directory>
#</VirtualHost>
```

Рисунок 17 – httpd-vhosts.conf

Был установлен WordPress, а распакованный архив был перенесен по пути “C:/xampp/htdocs/test_site”. Также была настроена база данных. После этого был осуществлен переход на <http://test.site> и пройдена регистрация.

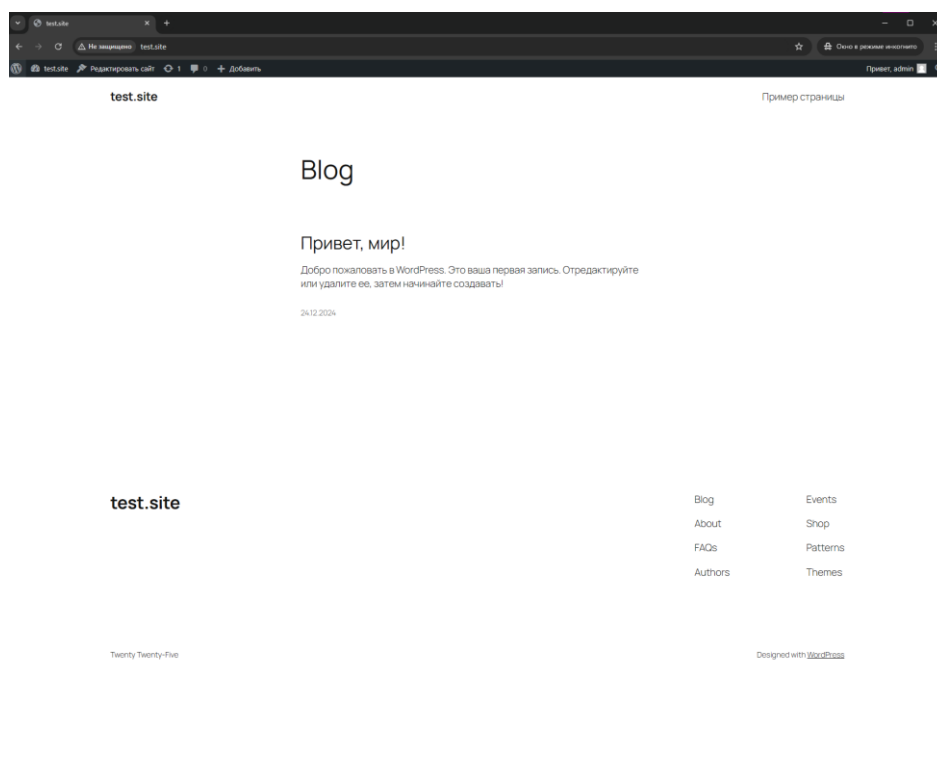


Рисунок 18 - <http://test.site>

Была изменена тема.

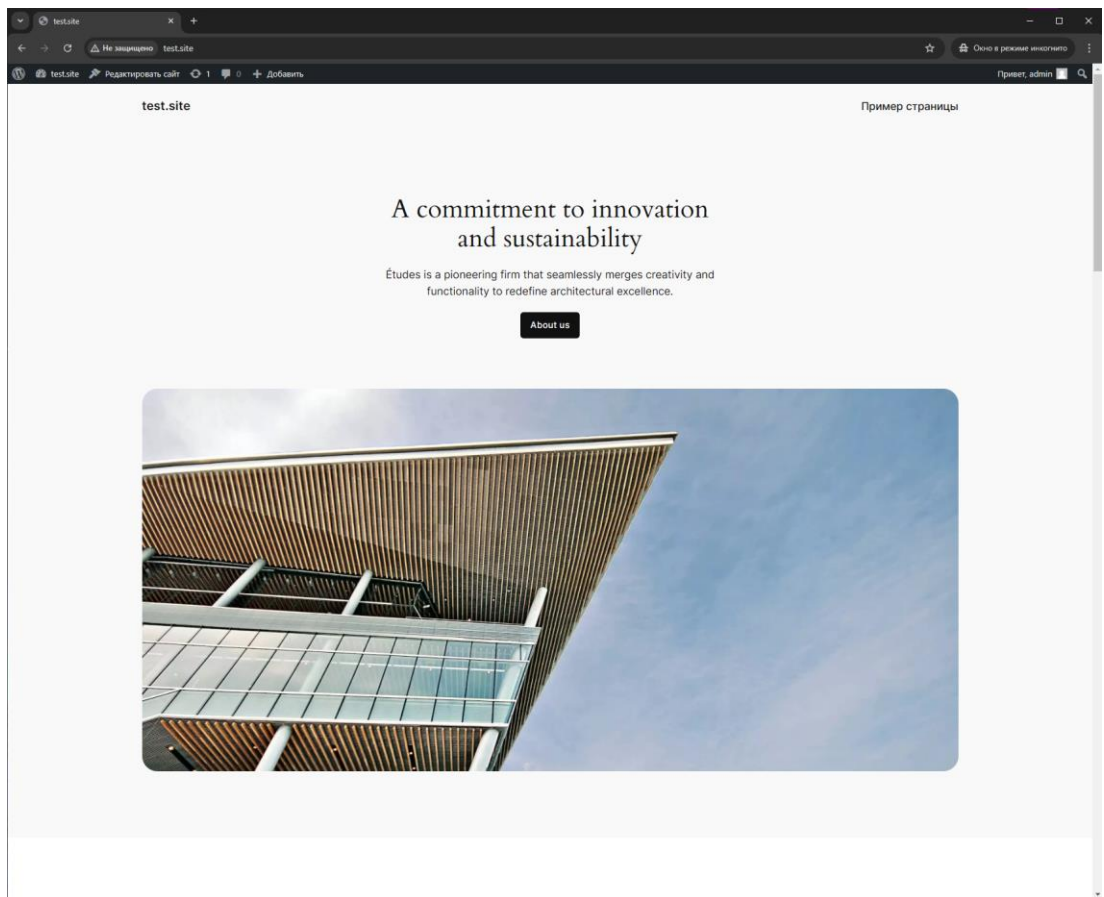


Рисунок 19 - <http://test.site> с измененной темой

Заключение

В рамках данной работы были изучен инструмент gulp, а также создан простейший сайт с использованием инструментов XAMPP и Wordpress