

**Министерство науки и высшего образования Российской Федерации**  
**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ**  
**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**  
**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**  
**ITMO University**

**ОТЧЁТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №3**

**По дисциплине** Web-программирование

**Обучающийся** Зорина Яна Сергеевна

**Факультет** Факультет инфокоммуникационных технологий

**Группа** K3322

**Направление подготовки** 11.03.02 Инфокоммуникационные технологии и  
системы связи

**Образовательная программа** Программирование в инфокоммуникационных  
системах

<b>Обучающийся</b>	<u>16.01.2024</u> (дата)	<u>                    </u> (подпись)	<u>Зорина Я.С.</u> (Ф.И.О.)
<b>Руководитель</b>	<u>                    </u> (дата)	<u>                    </u> (подпись)	<u>Марченко Е.В.</u> (Ф.И.О.)

Санкт Петербург

2024

## **СОДЕРЖАНИЕ**

Введение.....	3
Задание 1 .....	4
1.    Пункт А .....	4
2.    Пункт Б.....	5
Задание 2 .....	7
Заключение .....	10

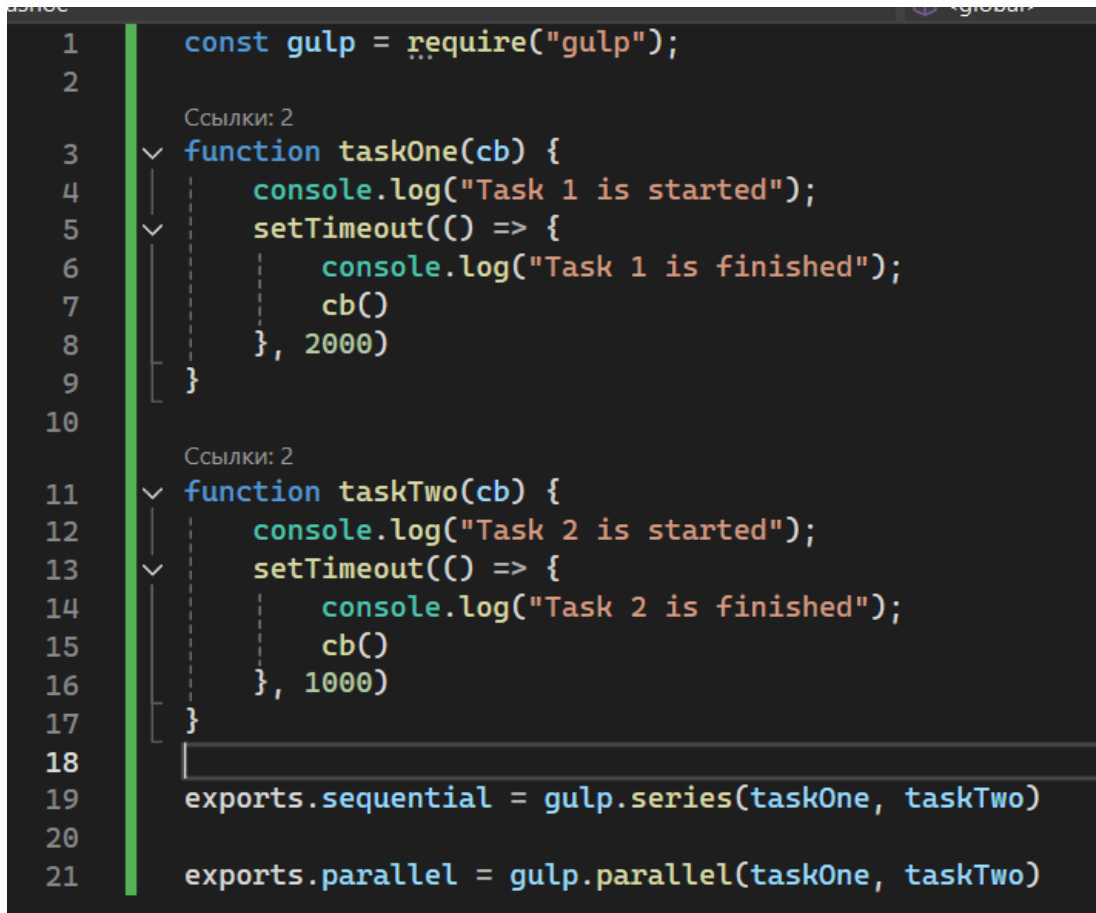
## **ВВЕДЕНИЕ**

Целью практической работы является освоение работы с gulp, создание формы для отправки информации по обратной связи, установка инструментария для отладки проектов и настройка портала test.site.

## ЗАДАНИЕ 1

### 1. Пункт А

В данном пункте требуется создать два таска и настроить их на последовательное и параллельное выполнение. На рисунке представлен скрипт `gulpfile.js`, где можно заметить искусственную задержку (она сделана для наглядности выполнения).



```
1  const gulp = require("gulp");
2
3  function taskOne(cb) {
4    console.log("Task 1 is started");
5    setTimeout(() => {
6      console.log("Task 1 is finished");
7      cb()
8    }, 2000)
9  }
10
11 function taskTwo(cb) {
12   console.log("Task 2 is started");
13   setTimeout(() => {
14     console.log("Task 2 is finished");
15     cb()
16   }, 1000)
17 }
18
19 exports.sequential = gulp.series(taskOne, taskTwo)
20
21 exports.parallel = gulp.parallel(taskOne, taskTwo)
```

Чтобы запустить таски последовательно необходимо прописать в консоли команду `gulp sequential`, параллельно – `gulp parallel`. На скрине ниже представлено выполнение команд.

```

PS D:\выбирай итмо и не выбирай вообще\5 сем\веб\lab_3\lab_3> gulp sequential
[12:33:14] Using gulpfile D:\выбирай итмо и не выбирай вообще\5 сем\веб\lab_3\lab_3\gulpfile.js
[12:33:14] Starting 'sequential'...
[12:33:14] Starting 'taskOne'...
Task 1 is started
Task 1 is finished
[12:33:16] Finished 'taskOne' after 2.01 s
[12:33:16] Starting 'taskTwo'...
Task 2 is started
Task 2 is finished
[12:33:17] Finished 'taskTwo' after 1.01 s
[12:33:17] Finished 'sequential' after 3.02 s
PS D:\выбирай итмо и не выбирай вообще\5 сем\веб\lab_3\lab_3> gulp parallel
[12:33:21] Using gulpfile D:\выбирай итмо и не выбирай вообще\5 сем\веб\lab_3\lab_3\gulpfile.js
[12:33:21] Starting 'parallel'...
[12:33:21] Starting 'taskOne'...
[12:33:21] Starting 'taskTwo'...
Task 1 is started
Task 2 is started
Task 2 is finished
[12:33:22] Finished 'taskTwo' after 1.01 s
Task 1 is finished
[12:33:23] Finished 'taskOne' after 2 s
[12:33:23] Finished 'parallel' after 2.01 s
PS D:\выбирай итмо и не выбирай вообще\5 сем\веб\lab_3\lab_3>

```

В случае последовательного выполнения сначала начинается и заканчивается первая задача, но при параллельном выполнении одновременно начинаются две задачи, но вторая заканчивается быстрее из-за меньшей задержки.

## 2. Пункт Б

В данном пункте задания требуется настроить отображения файлов проекта в браузере и его перезагрузки при изменении файлов. Для этого создается экземпляр `browserSync`, а в `gulpfile.js` добавляется еще одна задача.

В качестве директории, обслуживаемой сервером, указывается нынешняя. Функция `watchFiles` говорит просматривать все файлы директории с указанными расширениями и вызывать перезагрузку сервера при изменении любого из них. Конечный скрипт представлен ниже:

```
const browserSync = require("browser-sync").create()
```

1 ссылка

```
function serve(cb) {  
  browserSync.init({  
    server: {  
      baseDir: "./",  
    },  
  });  
  cb();  
}
```

1 ссылка

```
function watchFiles(cb) {  
  gulp.watch("*.html").on("change", browserSync.reload);  
  gulp.watch("css/*.css").on("change", browserSync.reload);  
  gulp.watch("js/*.js").on("change", browserSync.reload);  
  cb();  
}
```

```
exports.serve = gulp.series(serve, watchFiles);
```

## ЗАДАНИЕ 2

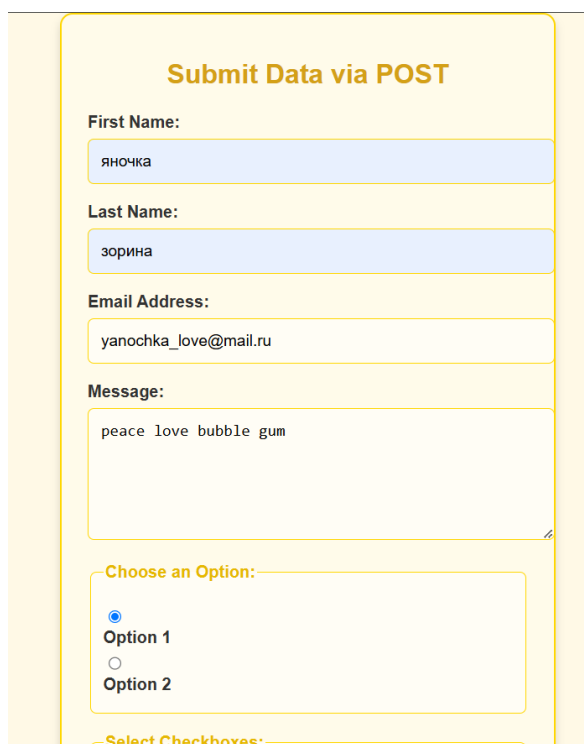
В данном задании требовалось создать форму для отправки обратной связи с радиокнопками и флажками. Поскольку для получения записанных данных нужен только метод POST, было принято решение записывать данные в файл, а затем (при желании) их демонстрировать. Для этого понадобится метод GET.

В файле `save_data.php` мы проверяем, что запрос к скрипту выполнен методом POST, после чего получаем данные из одноименного массива, записывая в переменные. Далее генерируется уникальный id, создается переменная, содержащая путь до файла, после чего данные сохраняются по заданному пути в нужном формате. В конце в новом окне выводится информация о том, что данные были успешно отправлены.

В файле `get_data.php` проверяем, что запрос выполнен через GET и передан id. Далее id записывается в переменную и ищется файл с соответствующим именем. Содержимое файла считывается, декодируется в JSON-массив, после чего выводится на страницу.

К сожалению, php-файлы получились слишком длинными, чтобы поместиться на скрине, поэтому при желании с ними можно ознакомиться в пул реквесте.

Результат представлен на скринах ниже.



The screenshot shows a web form with a yellow background and a white border. The title "Submit Data via POST" is centered at the top in orange. Below the title are several input fields: "First Name:" with the value "яночка", "Last Name:" with the value "зорина", "Email Address:" with the value "yanochka\_love@mail.ru", and "Message:" with the value "peace love bubble gum". At the bottom, there is a section "Choose an Option:" with two radio buttons, "Option 1" (selected) and "Option 2". Below this is a section "Select Checkboxes:" which is partially visible.

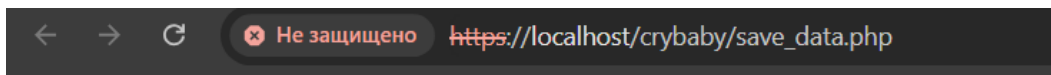
**Choose an Option:**

☒ Option 1  
☐ Option 2

**Select Checkboxes:**

☐ Checkbox 1  
☒ Checkbox 2  
☒ Checkbox 3

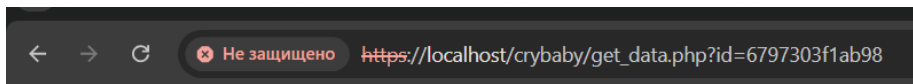
Submit Data



## Data saved successfully!

Record ID: 6797303f1ab98

Retrieve Data



## Data for ID: 6797303f1ab98

**First Name:** яночка

**Last Name:** зорпина

**Email:** yanochka\_love@mail.ru

**Message:** peace love bubble gum

**Selected Option:** option1

**Selected Checkboxes:**

- option2
- option3

**Submission Time:** 2025-01-27 08:05:35

Для запуска проекта использовался XAMPP. Название проекта: crybaby.



### ЗАДАНИЕ 3

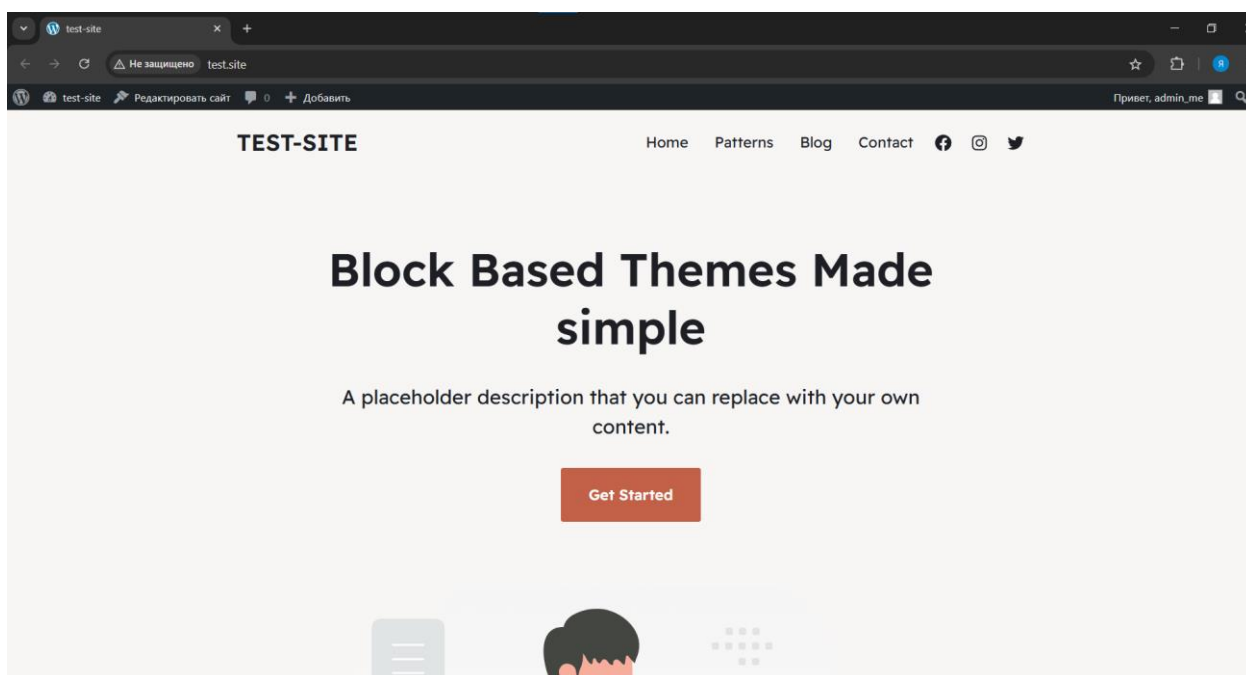
В данном задании требовалось установить инструментарий (использовала XAMPP) и движок wordpress с одноименных официальных сайтов.

Архив wordpress был распакован внутри XAMPP (D:\xampp\htdocs\test.site). На сайте <http://localhost/phpmyadmin> была создана база данных test-site с кодировкой utf8mb4\_general\_ci. В файле httpd-vhost.conf была добавлена следующая конфигурация:

```
43 <VirtualHost *:80>
44     DocumentRoot "D:\xampp\htdocs\test.site"
45     ServerName test.site
46     <Directory "D:\xampp\htdocs\test.site">
47         AllowOverride all
48         Require all granted
49     </Directory>
50 </VirtualHost>
51
```

В файле C:\Windows\System32\drivers\etc\hosts добавлена строка “127.0.0.1 test.site”.

Далее перейдя по адресу test.site была начата установка, указано имя бд и создана учетная запись. После создания, входа и настройки темы при переходе на сайт отображается следующая информация:



## **ЗАКЛЮЧЕНИЕ**

В данном отчёте по практической работе были выполнены упражнения и описаны результаты выполнения программы. Цели достигнуты.