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

Факультет Инфокоммуникационных Технологий

# Web-программирование

**Лабораторная работа 3**

Выполнил Ветошкин Р. В. Проверила Марченко Е. В.

Санкт-Петербург, 2024

**Цель:** научиться работать с gulp, создать форму для получения информации от клиента и научиться работать с Wordpress.

# Задачи:

* Научиться работать с gulp – создать таски, настроить параллельно и последовательно выполнение, а также настроить автоматическую перезагрузку при изменении файлов проекта
* Создать форму для приема обратной связи от клиента
* Установить движок Wordpress и запустить сайт

# Задание 1

Для выполнения данного задания был создан gulpfile с двумя тасками: styles для компиляции scss в css и scripts для объединения js файлов в один main.js, потом были созданы таски для выполнения их в последовательном (build-sequential) и параллельном режимах (build-parallel). Был создан task watch для отслеживания изменений и перезагрузки браузера.

Рисунок 1 – gulpfile.js

Рисунок 2 – gulpfile.js

Далее были написаны простые index.html, style.scss и file.js:

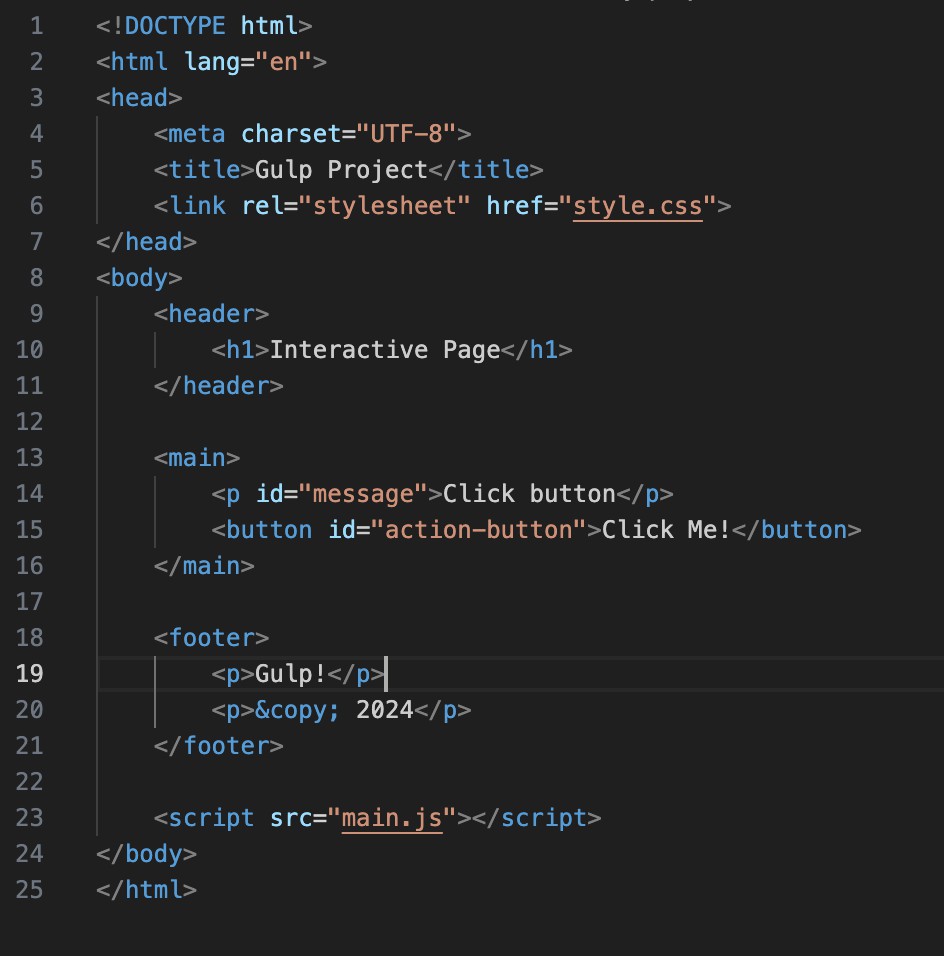


Рисунок 3 – index.html

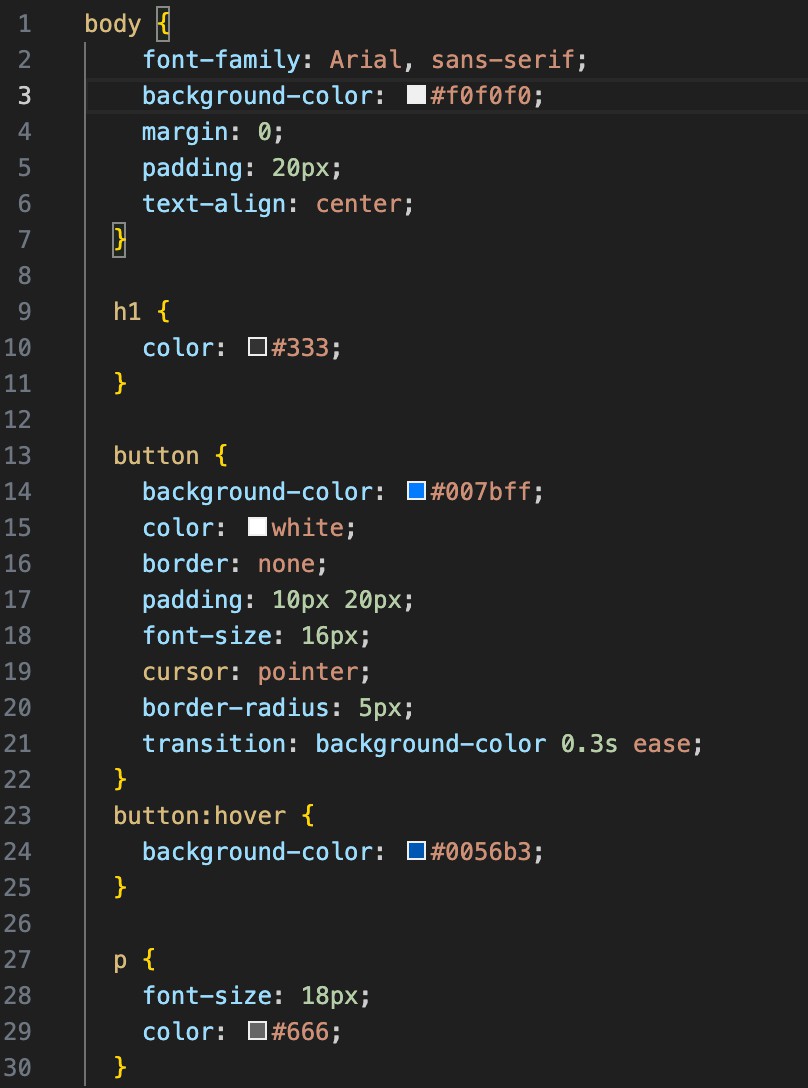


Рисунок 4 – style.scss

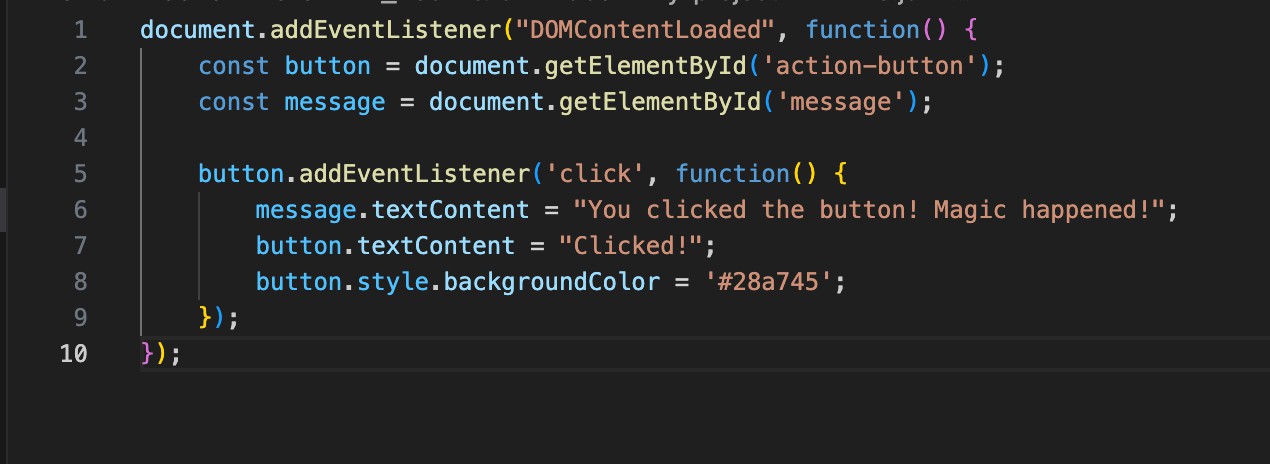


Рисунок 5 – file.js

Результат параллельного запуска:

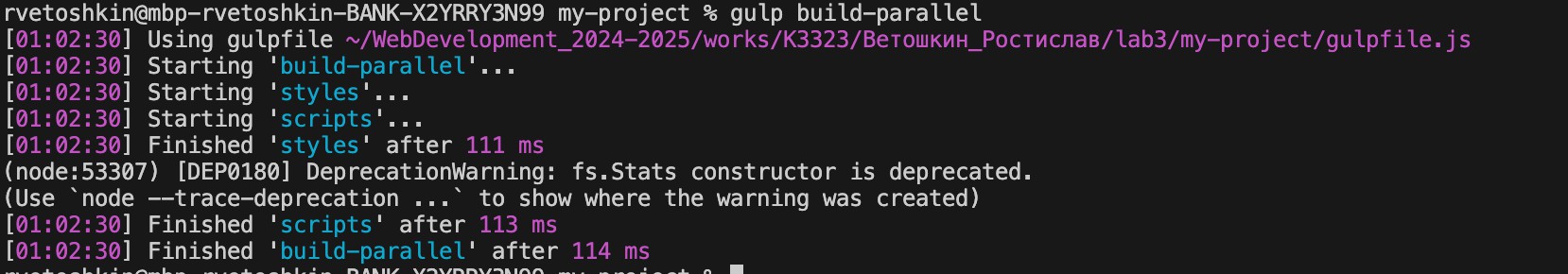


Рисунок 6 – параллельный запуск Результат последовательного запуска:

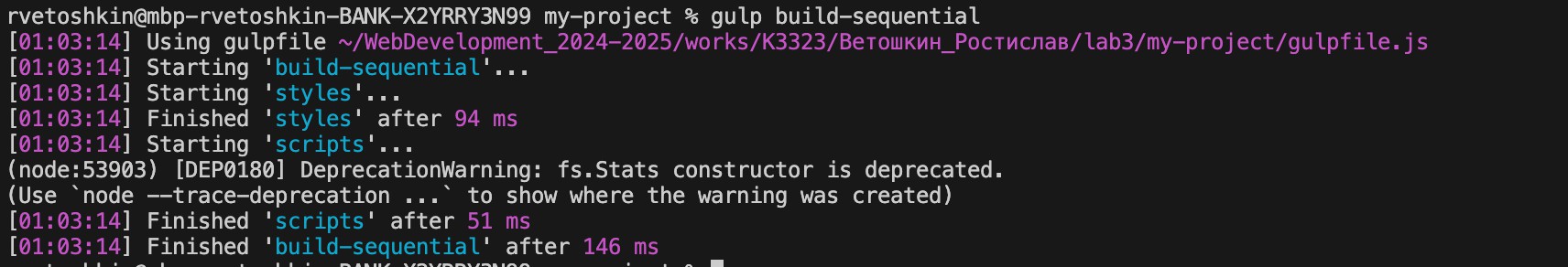


Рисунок 7 – последовательный запуск

В файлы проекта успешно добавились style.css и main.js:

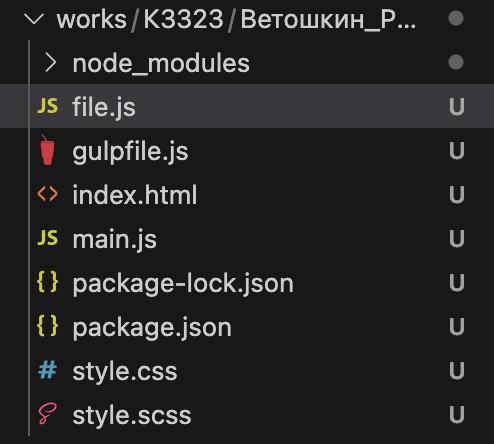


Рисунок 8 – папка проекта Потом был запущен browser-sync:



Рисунок 9 – запуск gulp с browser-sync

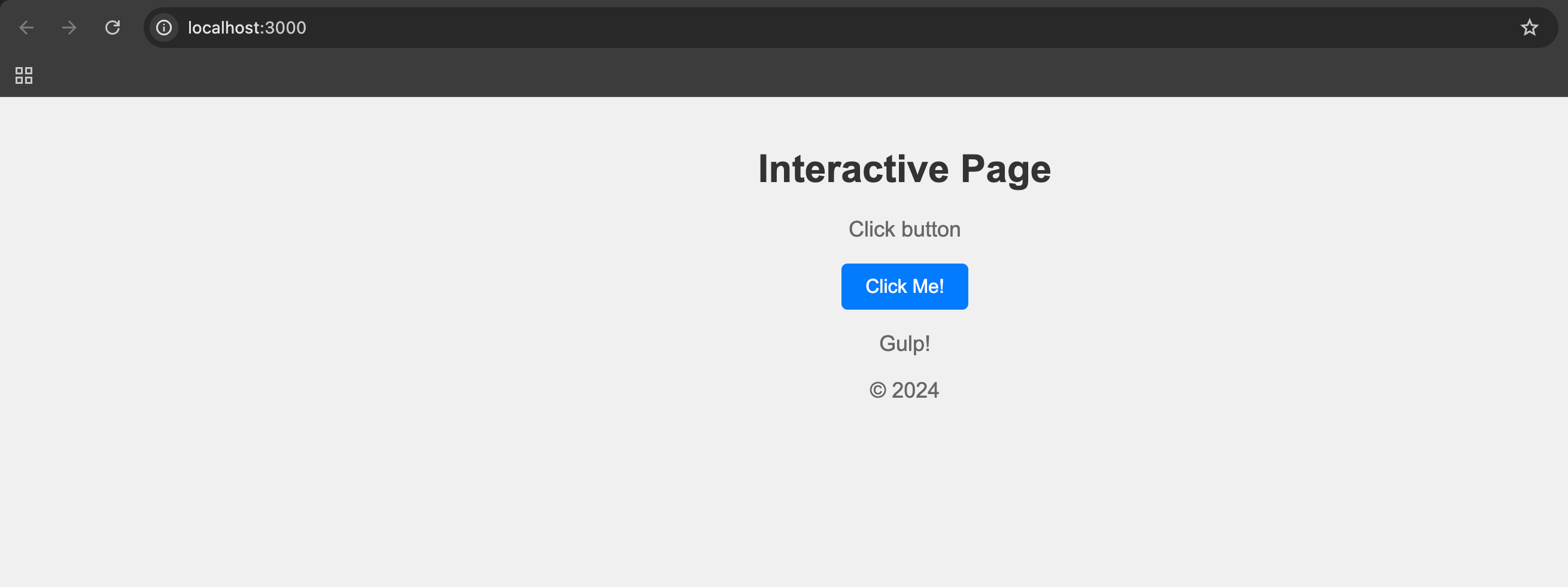
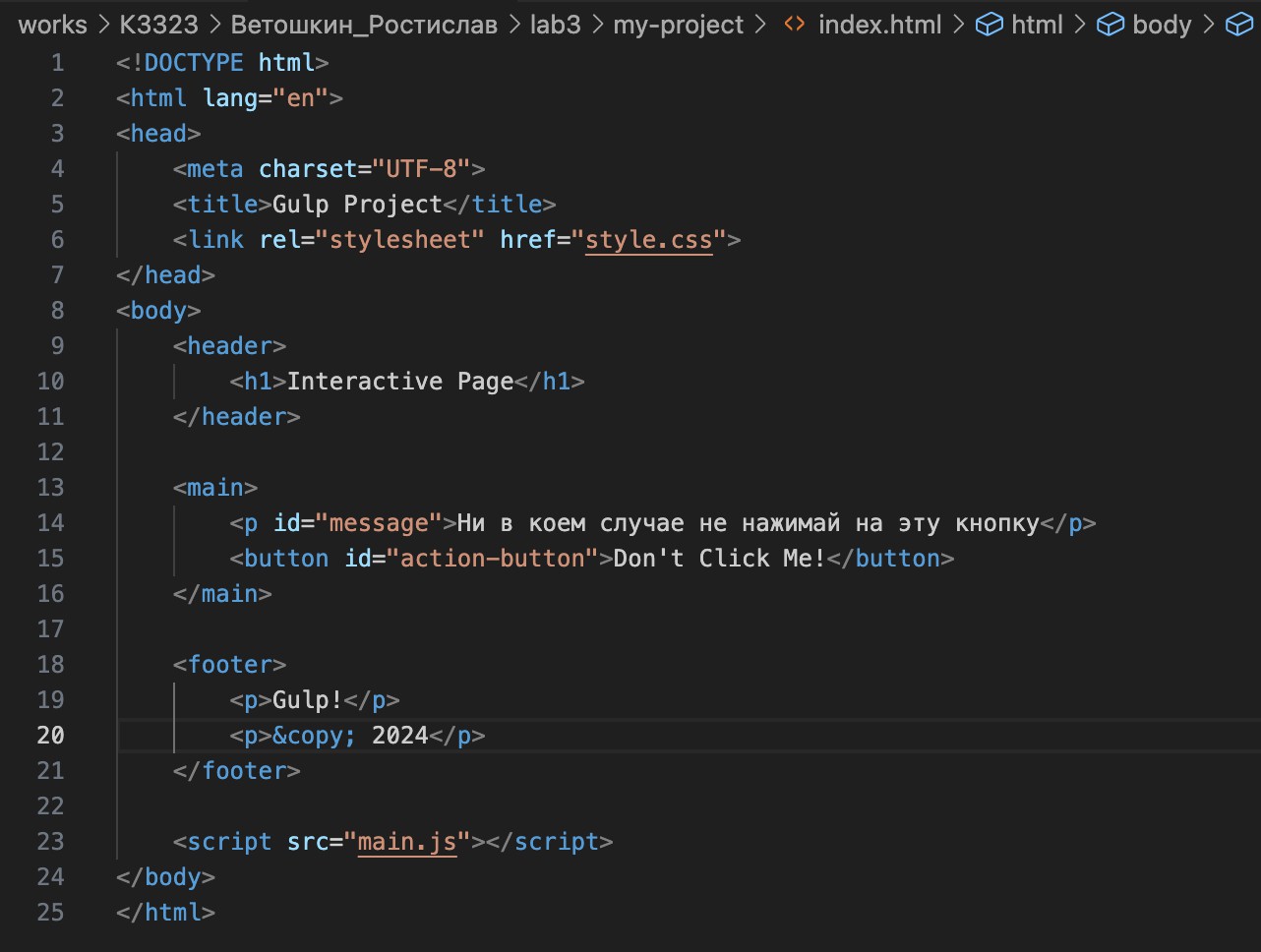


Рисунок 10 – результат запуска gulp с browser-sync

Как видно, при внесении изменений страница автоматически перезагрузилась и изменения отобразились:



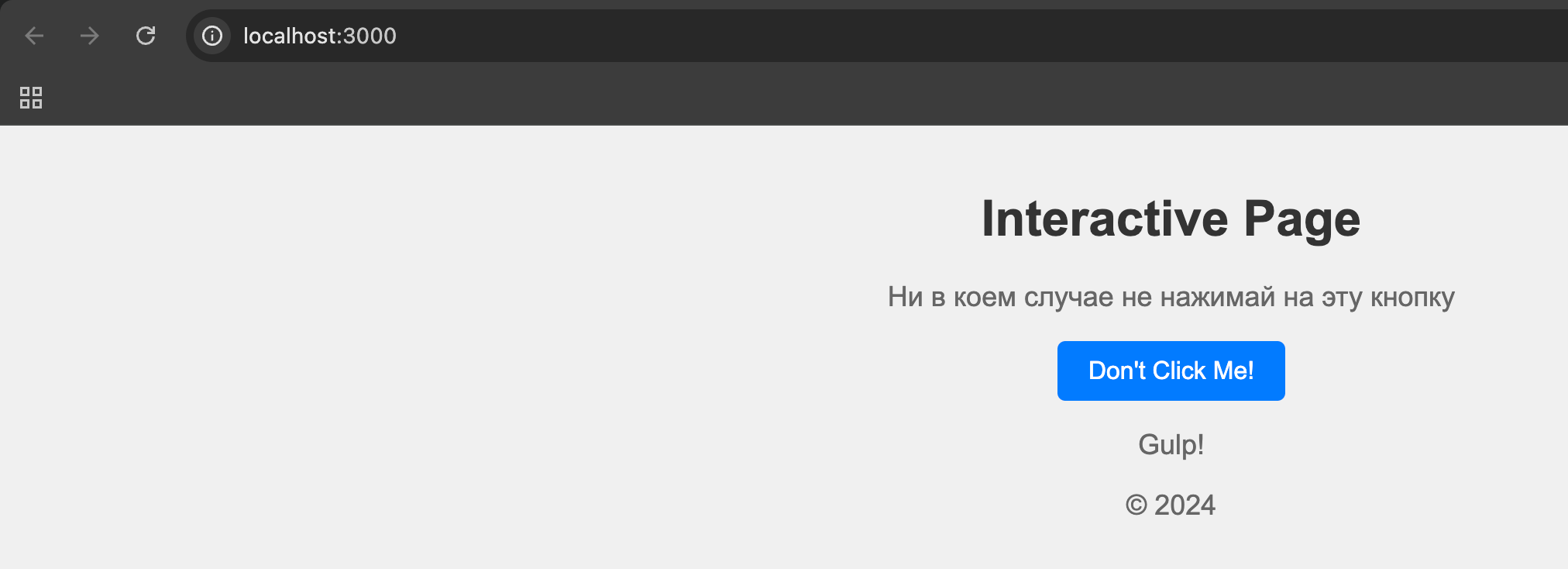
Рисунок 11 – изменения в index.html

Рисунок 12 – автоматическое обновление

# Задание 2

Сначала была написана простая форма, где можно отправлять и GET, и

POST запросы:

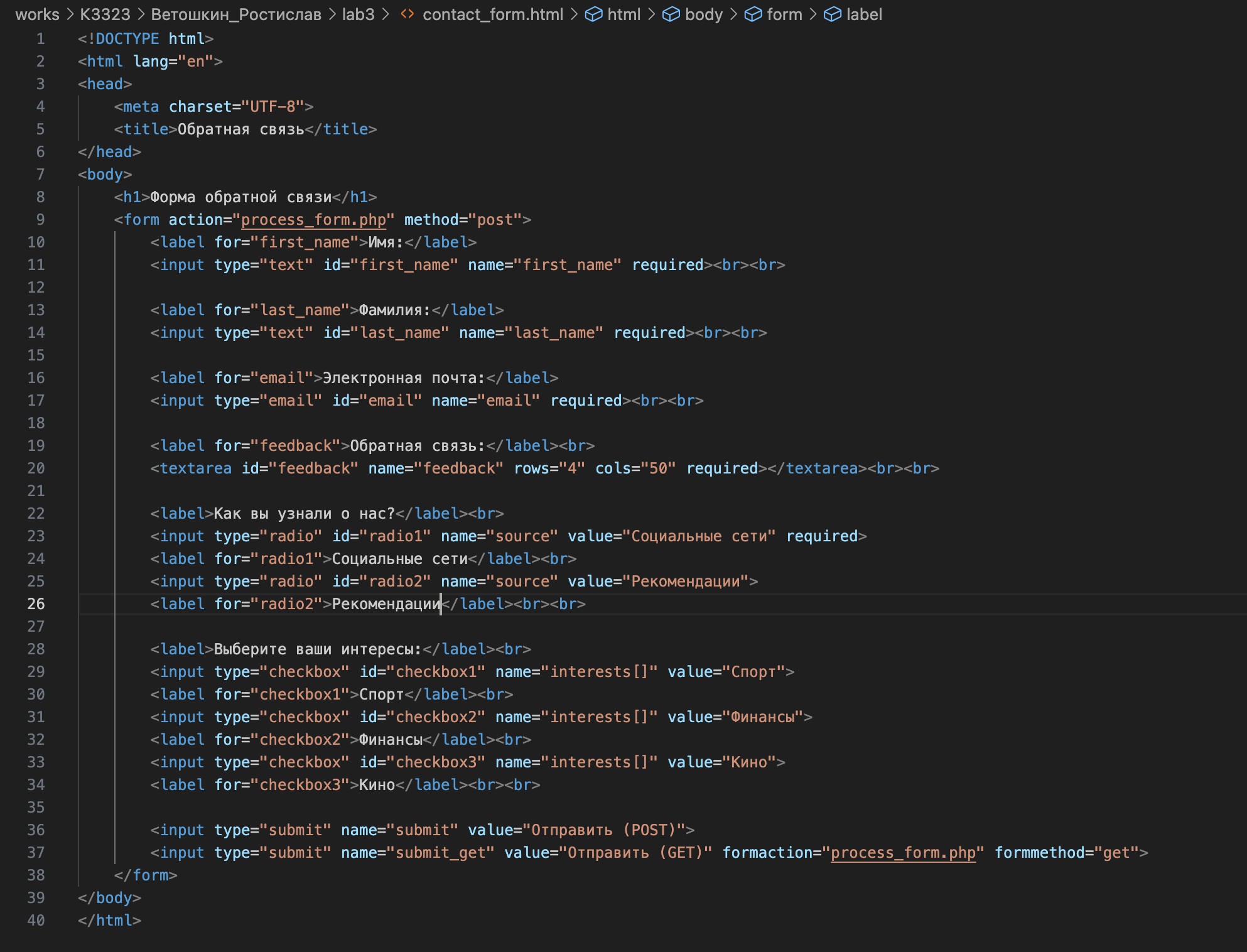


Рисунок 13 – код contact\_form.html

Потом был написан php файл для обработки и POST, и GET запросов:



Рисунок 14 – process\_form.php

Потом был запущен сервер:

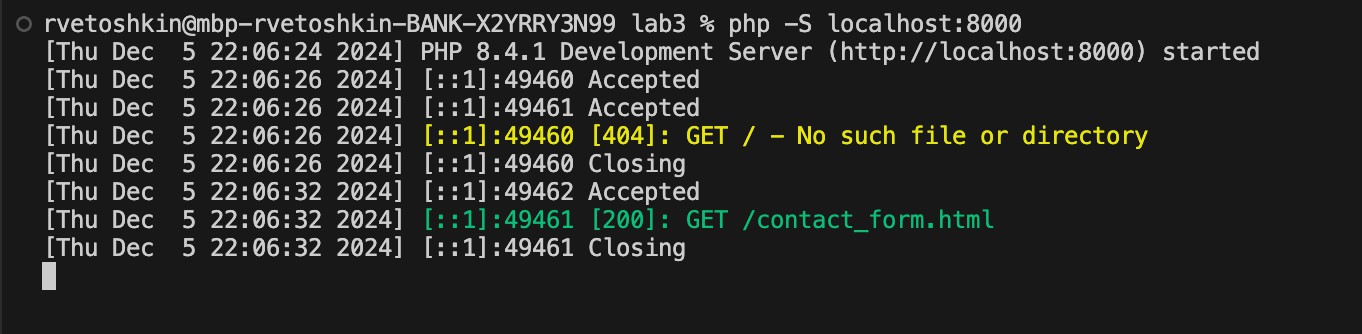


Рисунок 15 – запуск сервера При переходе открывается форма:

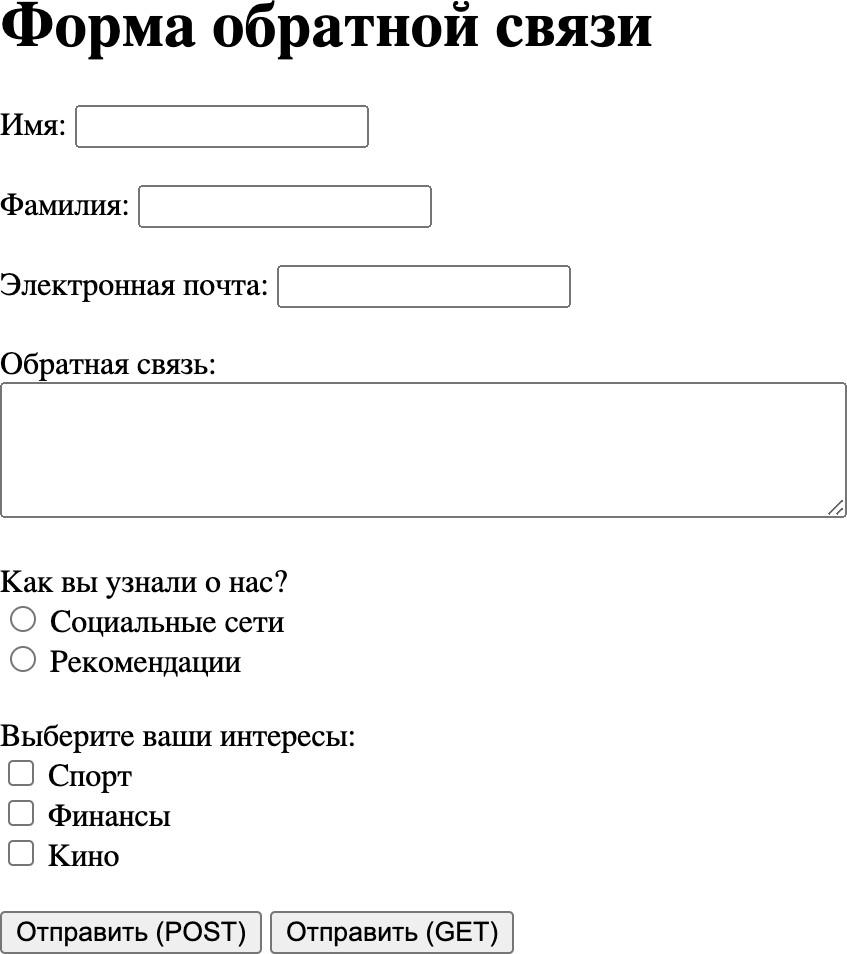


Рисунок 16 – contact\_form.html

Заполнение формы:

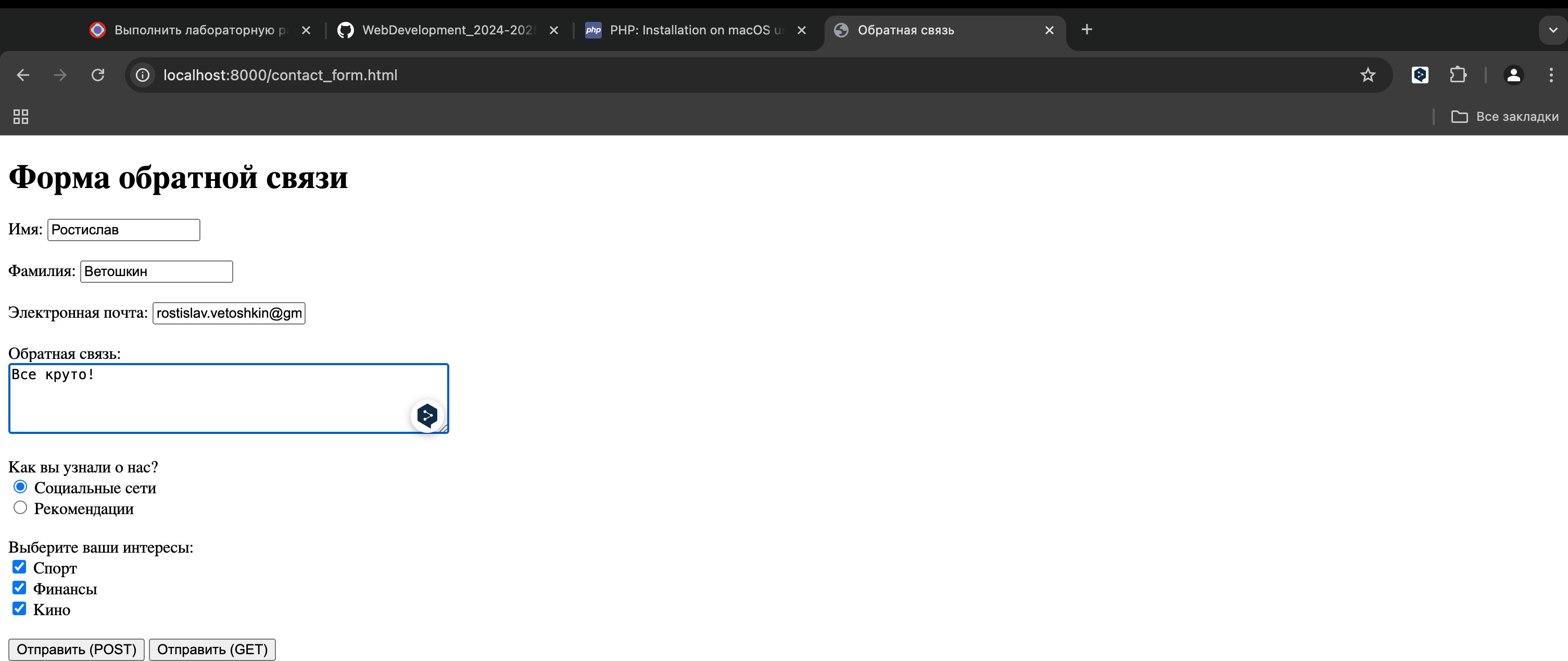


Рисунок 17 – contact\_form.html

Результат отправки POST:

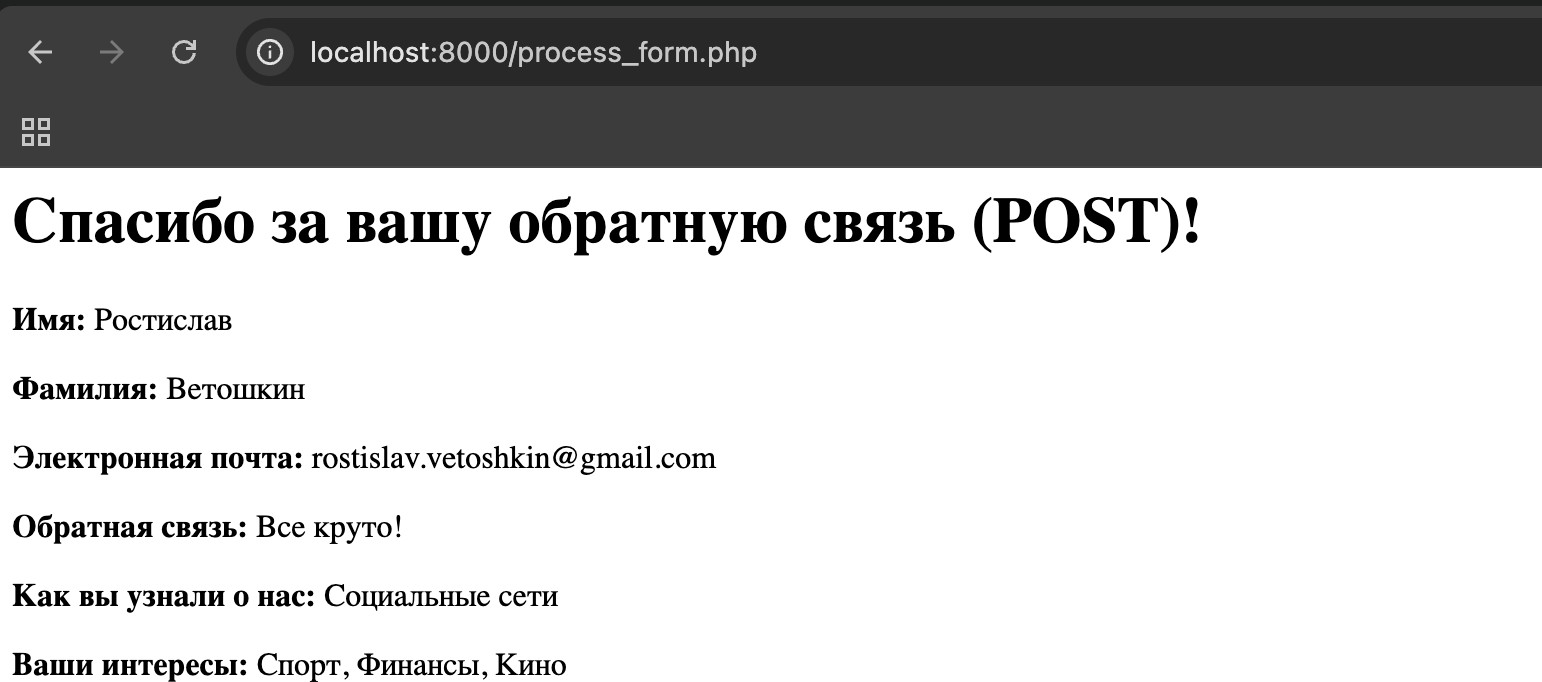


Рисунок 17 – process\_form.php

Результат отправки GET:

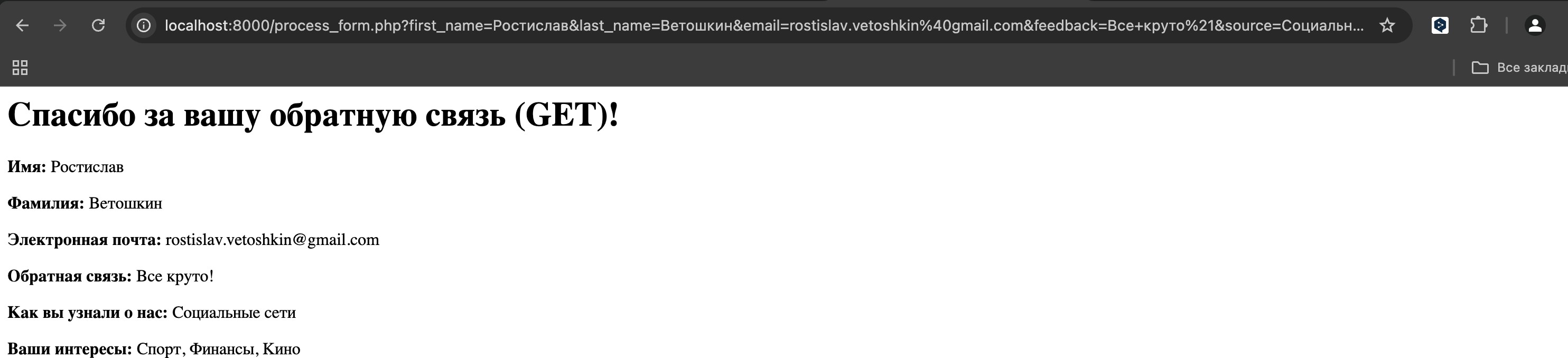


Рисунок 18 – process\_form.php

Метод GET передает данные в URL, подходит, когда данные не конфиденциальны, в данном случае такой URL получается:

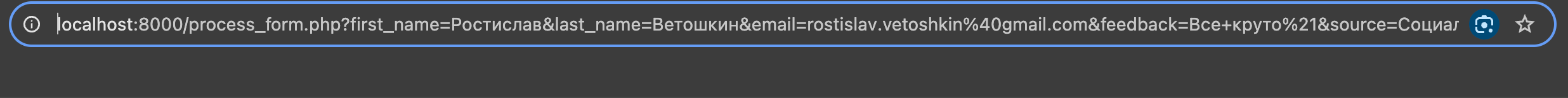


Рисунок 19 – URL



Рисунок 20 – URL

Метод POST передает данные в теле запроса, они не видны в URL, что позволяет передавать конфиденциальную информацию:

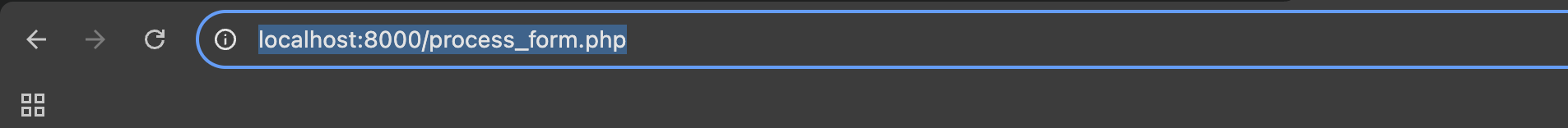


Рисунок 21 – URL

# Задание 3

Для установки WordPress был использован WAMP. Сначала были скачены движок WordPress и WAMP.

Потом была добавлена test\_db:

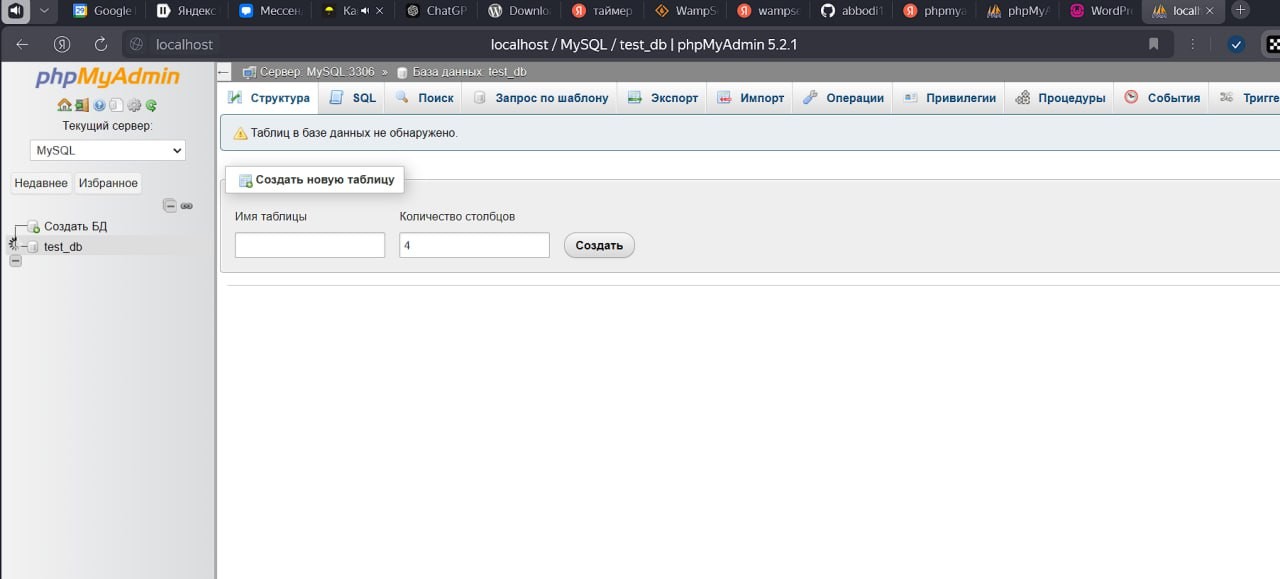


Рисунок 22 – test\_db

Потом был выполнен переход на <http://localhost/wordpress/> и была выполнена настройка WordPress:

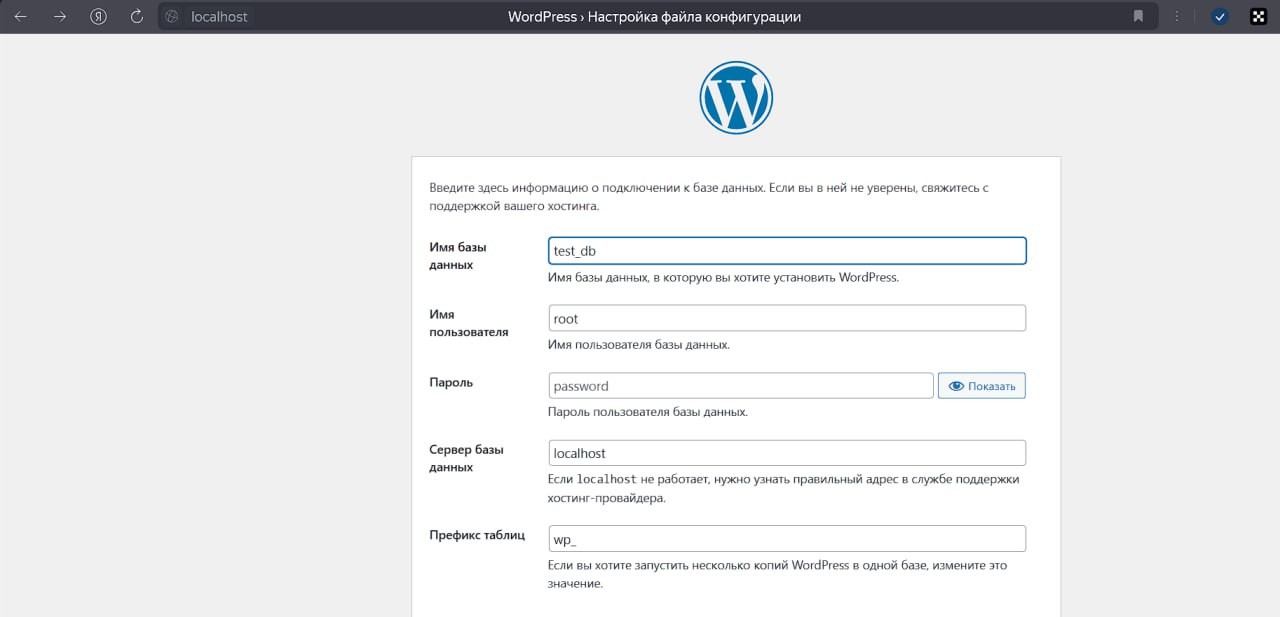


Рисунок 23 – wordpress

Чтобы был доступ через [http://test.site](http://test.site/) сначала была добавлена такая запись в файл hosts:

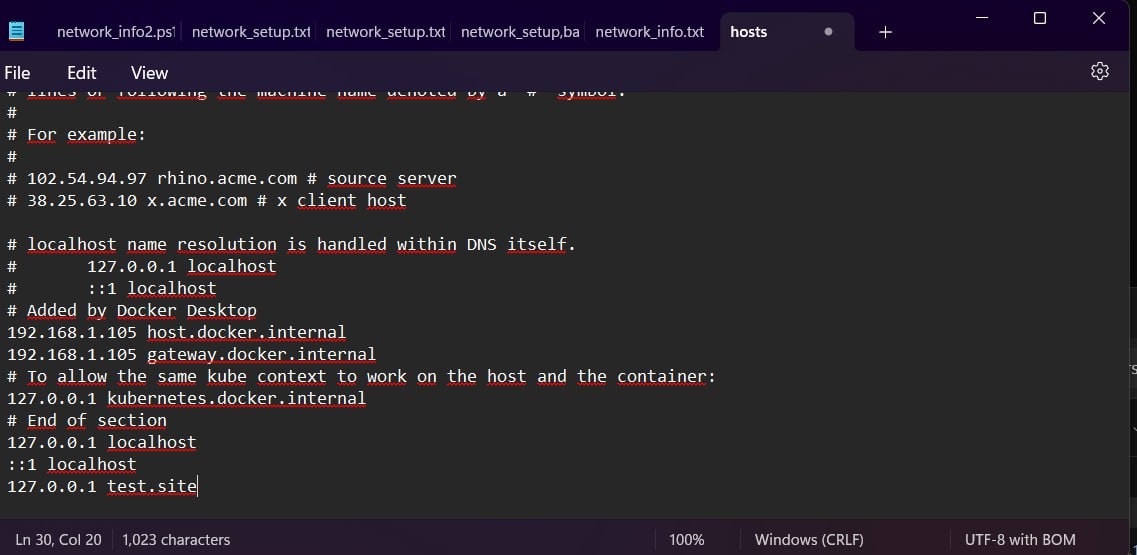


Рисунок 24 – файл hosts

Потом был добавлен такой виртуальный хост в Apache:

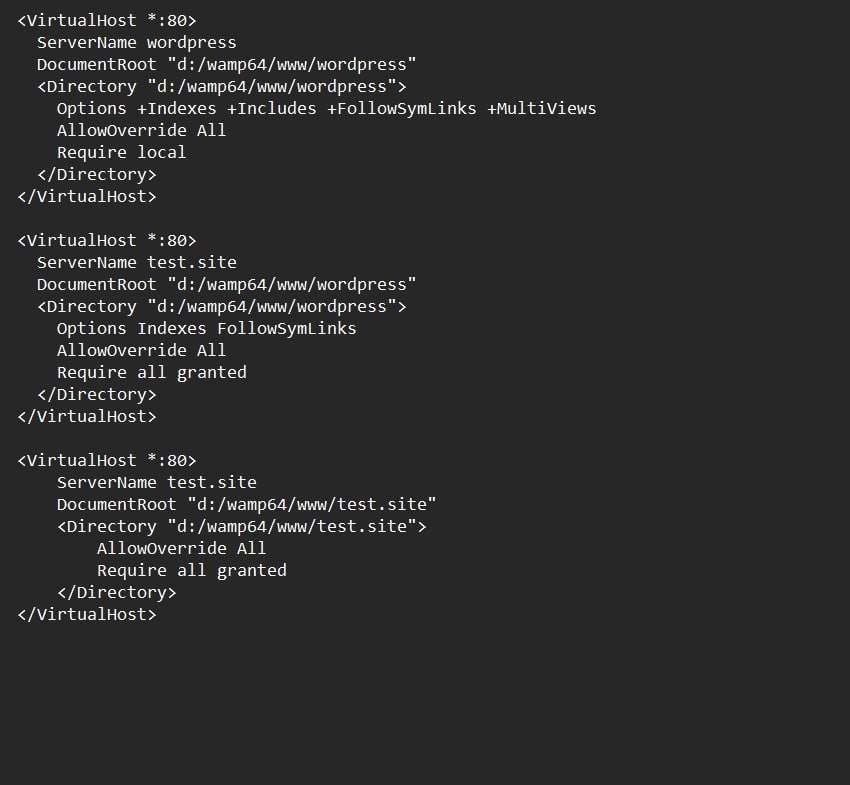


Рисунок 25 – файл httpd-vhosts.conf

В итоге при вводе [http://test.site](http://test.site/) отображается сайт:

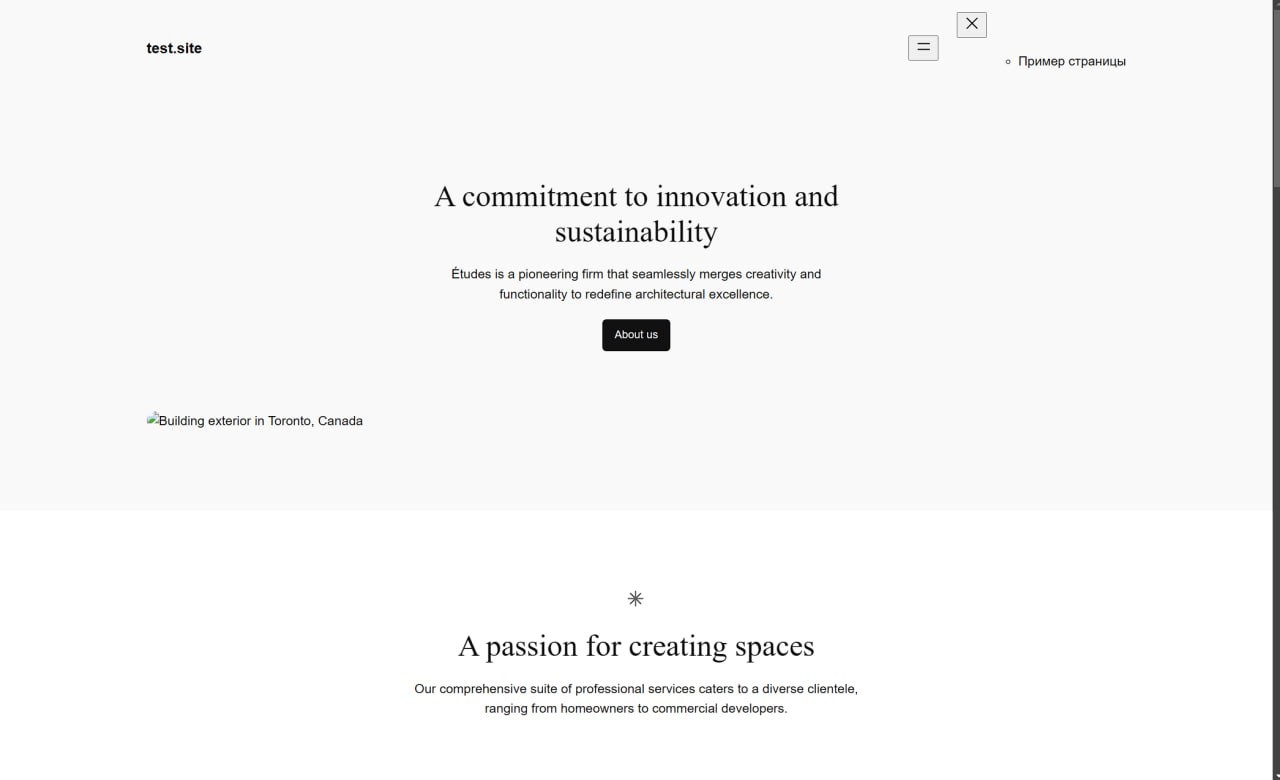


Рисунок 26 – переход на [http://test.site](http://test.site/)

**Вывод:** был изучен gulp, была создана форма обратной связи и был создан сайт на Wordpress.