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

ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ 2

По дисциплине Web-программирование

Тема работы Основы Git и Galp

Обучающийся Алексеев Тимофей Юрьевич

Факультет Факультет инфокоммуникационных технологий

Группа К3221

Направление подготовки 11.03.02 Инфокоммуникационные технологии и системы связи

Образовательная программа Программирование в инфокоммуникационных системах

Обучающийся	01.10.2024 (дата)	_____ (подпись)	Алексеев Т.Ю. (Ф.И.О.)
Руководитель	_____ (дата)	_____ (подпись)	Марченко Е.В. (Ф.И.О.)

Санкт-Петербург
2024 г.

Цель

Познакомиться с основами работы с Git и Gulp.

Задачи

1. Ознакомиться с основными запросами Git;
2. Создать репозиторий, сделать 3 коммита и синхронизировать;
3. Установить gulp;
4. Создать простую задачу с помощью gulp;
5. Создать переключатель веб-страниц с определенным интервалом с помощью gulp.

Ход работы

Задание 1

В данном задании было необходимо произвести работу с системой контроля версий Git.

Для этого в заранее созданной ветке lab_2 был добавлен код из лабораторной работы 1. После чего в каждый из файлов были внесены небольшие изменения. Далее был сделан коммит с помощью команд git add –all и git commit -m “N commit”.

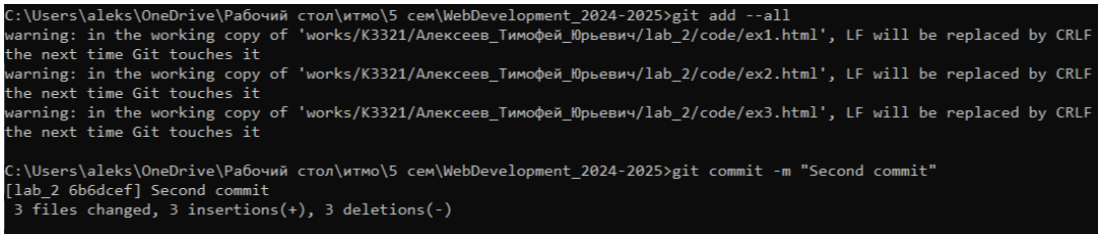

Рисунок 1 – Команды, с помощью которых создается коммит

Рефакторинг файлов был произведен два раза, в результате чего получилось сделать 3 коммита, которые можно посмотреть в истории репозитория.

Далее для синхронизации локального репозитория с удаленным была использована команда git push -u origin lab_2.

Рисунок 2 – Синхронизация удаленного репозитория и локального

В итоге текущий проект доступен по следующей ссылке: https://github.com/NorthPole0499/WebDevelopment_2024-2025/tree/lab_2

Задание 2

В данном упражнении требуется установить Gulp и создать task.

Для его установки пропишем в консоли следующие команды npm install gulp npm и install gulp-cli.

Далее проверим правильность установки путем запуска команды npx gulp –version. Получаем информацию об установленных версиях.

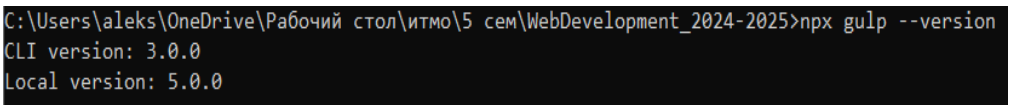

Рисунок 3 – Версия установленного gulp

Далее для создания первой задачи необходимо написать в консоль команду npm init, которая создает окружение.

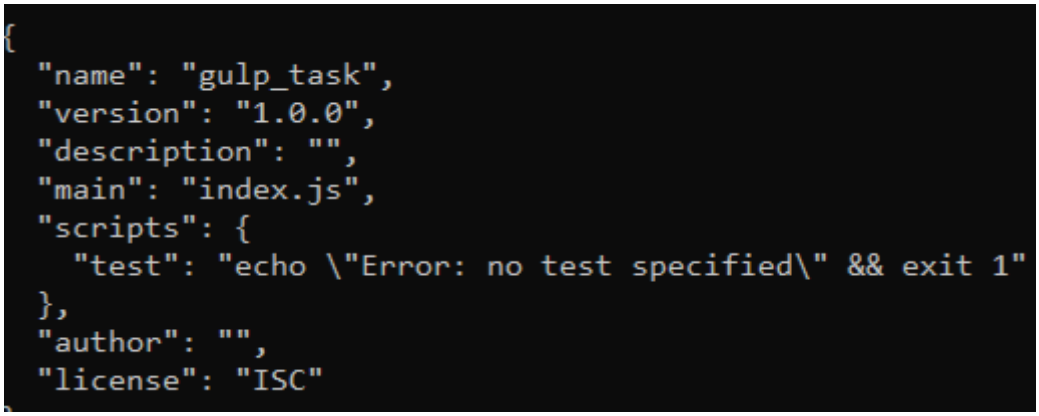

Рисунок 4 – Информация о текущем окружении

После этого устанавливаем в наше окружение gulp с помощью команды npm install gulp –save-dev. Теперь мы готовы к созданию первой задачи.

Создаем файл gulpfile.js в корневой папке нашего проекта. После чего прописываем там следующий код, обозначающий инициализирование задачи под именем hello.

```
var gulp = require('gulp');

gulp.task('hello', function() {
    console.log('Hello world, this code was written by Timofey!');
  });
```

Рисунок 5 – Код первой задачи Gulp

После этого добавляем в package.json в раздел scripts следующую команду под названием hello: gulp hello.

Прописав в консоли npm run hello, сможем увидеть результат выполнения первой задачи.

```
_task>npm run hello

> gulp_task@1.0.0 hello
> gulp hello

[20:18:06] Using gulpfile ~\OneDrive\Рабочий сто
ич\lab_2\gulp_task\gulpfile.js
[20:18:06] Starting 'hello'...
Hello world, this code was written by Timofey!
```

Рисунок 6 – Выполнение задачи gulp

Задание 3

В данном задании было необходимо создать в gulp задачу, которая сама бы переключала страницы в браузере с определенным интервалом.

Для решения данной задачи нам понадобится установка веб-сервера с помощью команды npm install browser-sync –save-dev.

Далее нам необходимо создать папку app, в которой будут храниться все исполняемые файлы. А именно index.html и script.js.

В index.html мы прописываем стандартный синтаксис. В head мы укажем css-стиль для тега iframe, который растянет его на весь экран. В body же вызовем сам тег iframe, отображающий другие веб-страницы, а также укажем наш script.js.

Рисунок 7 – Листинг кода файла index.html

После этого переходим к написанию script.js. В ней мы объявляем константы: адреса, интервал и тд. После этого создаем функцию ChangePage, которая присваивает атрибуту src у тега iframe адрес веб-страницы. После чего устанавливается вызов данной функции через указанный интервал.

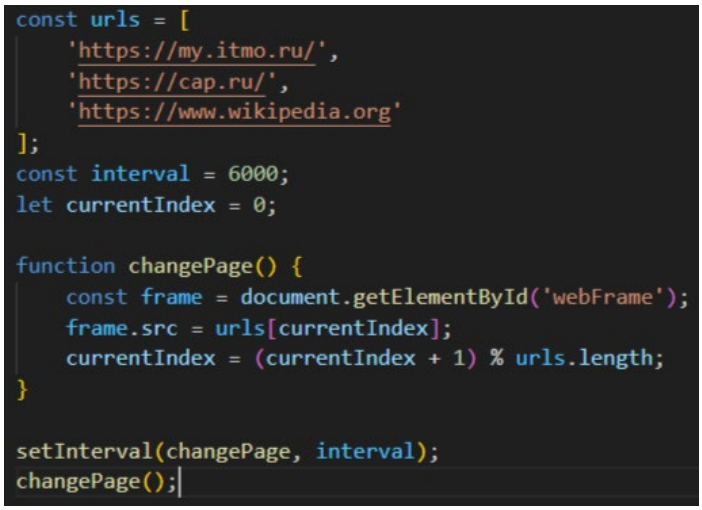

Рисунок 8 – Листинг кода файла script.js

После чего в файле gulpfiles.js создаем новую задачу, для которой используем наш веб-сервер с помощью browser-sync. При каждом изменении в каком-либо файле браузер будет перезагружаться.

Рисунок 9 – Задача opener

В итоге заносим нашу задачу в package.json под именем open: gult opener. После этого прописываем в консоль npm run open и наблюдаем, как в открывшейся вкладке переключаются страницы.

Вывод

В ходе выполнения лабораторной работы цель была достигнута. Было произведено ознакомление с основами работы с Git и Gulp.

В течение выполнения работы были изучены основные команды Git, создан репозиторий, сделаны коммиты и произведена синхронизация удаленного и локального репозиториев. Установлен gulp и написана первая задача, выводящая приветственное сообщение. Написана задача, которая вызывает веб-сервер, который переключает сайты из списка с определенным интервалом.