Звіт

Лабораторна робота №1

«Початкове налаштування web-проекту»

МЕТА: Опанувати практичними навичками початкового налаштування webпроекту з використанням таск-менеджера для автоматизації процесу webрозробки.

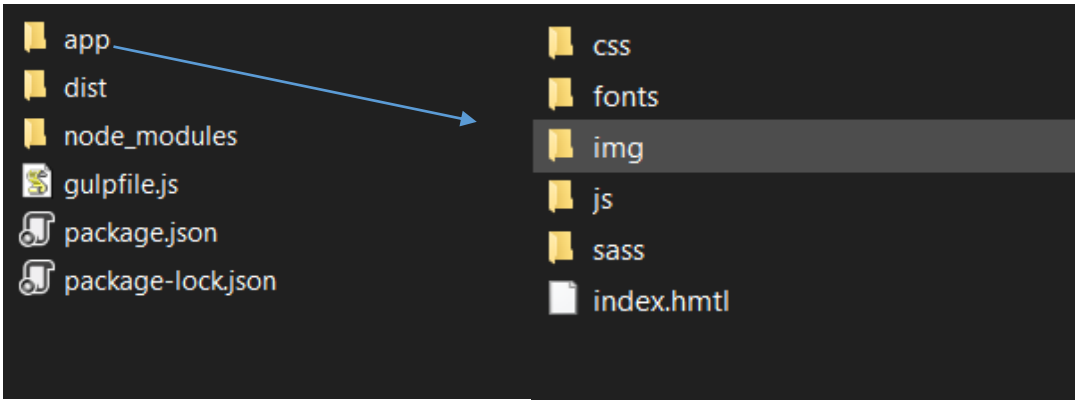
Перевірка версій встановлених програм:

```
C:\Users\roman>gulp -v
CLI version: 2.3.0
Local version: Unknown

C:\Users\roman>node -v
v14.17.0

C:\Users\roman>npm -v
6.14.13

C:\Users\roman>
```

Npm init

```
Press ^C at any time to quit.
package name: (myproject) package_lab1
version: (1.0.0)
description:
entry point: (index.js)
test command:
git repository:
keywords:
author: roman_shemenda
license: (ISC)
```

Структура папок та файлів :



Код gulpfile.js :

```
var gulp        = require('gulp'), // Подключаем Gulp
    sass        = require('gulp-sass'), //Подключаем Sass пакет,
    browserSync = require('browser-sync'), // Подключаем Browser Sync
    concat      = require('gulp-concat'), // Подключаем gulp-concat (для конкатенации файлов)
    uglify      = require('gulp-uglifyjs'), // Подключаем gulp-uglifyjs (для сжатия JS)
    cssnano     = require('gulp-cssnano'), // Подключаем пакет для минификации CSS
    rename      = require('gulp-rename'), // Подключаем библиотеку для переименования файлов
    del         = require('del'), // Подключаем библиотеку для удаления файлов и папок
    imagemin    = require('gulp-imagemin'), // Подключаем библиотеку для работы с изображениями
    pngquant    = require('imagemin-pngquant'), // Подключаем библиотеку для работы с png
    cache       = require('gulp-cache'), // Подключаем библиотеку кеширования
    autoprefixer = require('gulp-autoprefixer');// Подключаем библиотеку для автоматического добавления префиксов

gulp.task('sass', function() { // Создаем таск Sass
    return gulp.src('app/sass/**/*.sass') // Берем источник
        .pipe(sass()) // Преобразуем Sass в CSS посредством gulp-sass
        .pipe(autoprefixer(['last 15 versions', '> 1%', 'ie 8', 'ie 7'], { cascade: true })) // Создаем префиксы
        .pipe(gulp.dest('app/css')) // Выгружаем результата в папку app/css
        .pipe(browserSync.reload({stream: true})) // Обновляем CSS на странице при изменении
});

gulp.task('browser-sync', function() { // Создаем таск browser-sync
    browserSync({ // Выполняем browserSync
        server: { // Определяем параметры сервера
            baseDir: 'app' // Директория для сервера - app
        },
        notify: false // Отключаем уведомления
    });
});

gulp.task('scripts', function() {
    return gulp.src([ // Берем все необходимые библиотеки
        'app/libs/jquery/dist/jquery.min.js', // Берем jQuery
        'app/libs/magnific-popup/dist/jquery.magnific-popup.min.js' // Берем Magnific Popup
        ])
        .pipe(concat('libs.min.js')) // Собираем их в кучу в новом файле libs.min.js
        .pipe(uglify()) // Сжимаем JS файл
        .pipe(gulp.dest('app/js')); // Выгружаем в папку app/js
});

```

```
gulp.task('code', function() {
    return gulp.src('app/*.html')
    .pipe(browserSync.reload({ stream: true }))
});

gulp.task('css-libs', function() {
    return gulp.src('app/css/libs.sass') // Выбираем файл для минификации
        .pipe(sass()) // Преобразуем Sass в CSS посредством gulp-sass
        .pipe(cssnano()) // Сжимаем
        .pipe(rename({suffix: '.min'})) // Добавляем суффикс .min
        .pipe(gulp.dest('app/css')); // Выгружаем в папку app/css
});

gulp.task('clean', async function() {
    return del.sync('dist'); // Удаляем папку dist перед сборкой
});

gulp.task('img', function() {
    return gulp.src('app/img/**/*') // Берем все изображения из app
        .pipe(cache(imagemin({ // С кешированием
        // .pipe(imagemin({ // Сжимаем изображения без кеширования
            interlaced: true,
            progressive: true,
            svgoPlugins: [{removeViewBox: false}],
            use: [pngquant()]
        }))/**/)
        .pipe(gulp.dest('dist/img')); // Выгружаем на продакшен
});

gulp.task('prebuild', async function() {

```

```
72      var buildCss = gulp.src([ // Переносим библиотеки в продакшен
73          'app/css/main.css',
74          'app/css/libs.min.css'
75          ])
76      .pipe(gulp.dest('dist/css'))
77
78      var buildFonts = gulp.src('app/fonts/**/*') // Переносим шрифты в продакшен
79      .pipe(gulp.dest('dist/fonts'))
80
81      var buildJs = gulp.src('app/js/**/*') // Переносим скрипты в продакшен
82      .pipe(gulp.dest('dist/js'))
83
84      var buildHtml = gulp.src('app/*.html') // Переносим HTML в продакшен
85      .pipe(gulp.dest('dist'));
86
87  });
88
89  gulp.task('clear', function (callback) {
90      return cache.clearAll();
91  })
92
93  gulp.task('watch', function() {
94      gulp.watch('app/sass/**/*.sass', gulp.parallel('sass')); // Наблюдение за sass файлами
95      gulp.watch('app/*.html', gulp.parallel('code')); // Наблюдение за HTML файлами в корне проекта
96      gulp.watch(['app/js/common.js', 'app/libs/**/*.js'], gulp.parallel('scripts')); // Наблюдение за главным JS файлом и за библиотеками
97  });
98  gulp.task('default', gulp.parallel('css-libs', 'sass', 'scripts', 'browser-sync', 'watch'));
99  gulp.task('build', gulp.parallel('prebuild', 'clean', 'img', 'sass', 'scripts'));
```