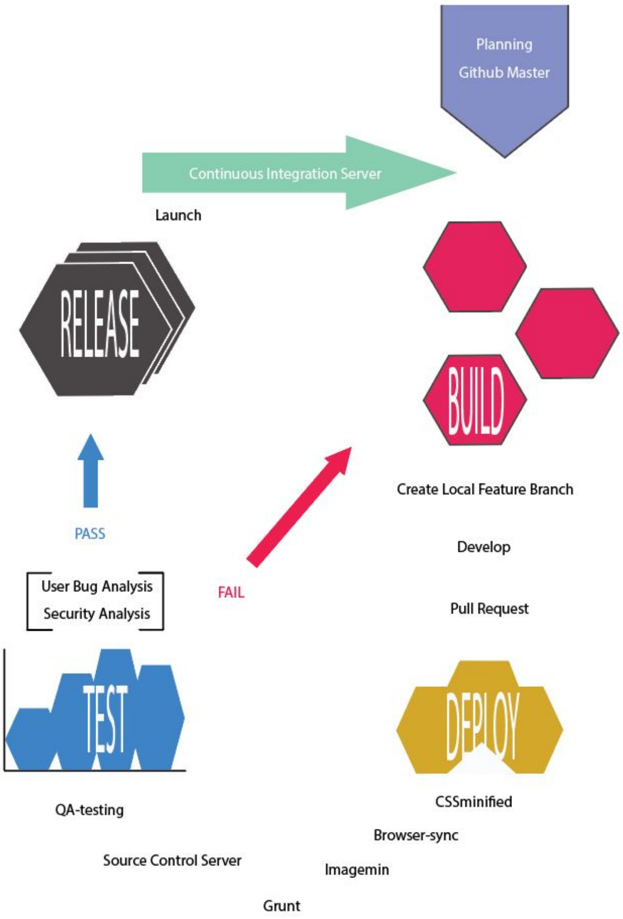
← Link to Theory.

Case Study

Your company has just implemented DevOps and has tasked you to set up the process with the development team and the operations teams. You have been asked to create a mobile application which records a person giving a speech. This should convert the speech to text and then upload it to a website.

Questions

1. Draw up your own process flow of the DevOps cycle which will be used in your company. Remember to include all the steps as well as the tools you will use.



2. Discuss 2 of these tools which you would use.

Minifying CSS - makes the css files smaller and the website faster.

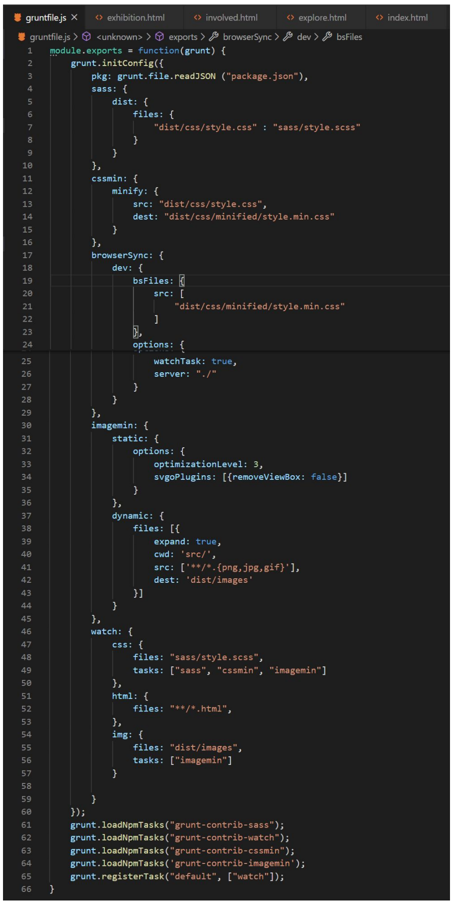
Browser-sync - is looking for changes in my css -and html-files to re-render the browser for me.

Link to Practical.

Practical

For this practical exercise you will be required to automate the assignment that you created for your HTML and CSS course in first year. You may use one of Grunt, Gulp or Webpack for the process. You may use Sass or Less.

- Please copy all HTML files from your app folder to a dist folder.
- The Sass or Less should be converted to a .css file and this should be minified.
- The images should be compressed from the original size to a smaller size by means of the task runner.
- The browser should reload when any file is changed.


gruntfile.js ×
exhibition.html
involved.html
explore.html
index.html
gruntfile.js > <unknown> > exports > browserSync > dev > bsFiles

```
module.exports = function(grunt) {
    grunt.initConfig({
        pkg: grunt.file.readJSON ("package.json"),
        sass: {
            dist: {
                files: {
                    "dist/css/style.css" : "sass/style.scss"
                }
            }
        },
        cssmin: {
            minify: {
                src: "dist/css/style.css",
                dest: "dist/css/minified/style.min.css"
            }
        },
        browserSync: {
            dev: {
                bsFiles: {
                    src: [
                        "dist/css/minified/style.min.css"
                    ]
                },
                options: {
                    watchTask: true,
                    server: "./"
                }
            }
        },
        imagemin: {
            static: {
                options: {
                    optimizationLevel: 3,
                    svgoPlugins: [{removeViewBox: false}]
                }
            },
            dynamic: {
                files: [{
                    expand: true,
                    cwd: 'src/',
                    src: ['**/*.{png,jpg,gif}'],
                    dest: 'dist/images'
                }]
            }
        },
        watch: {
            css: {
                files: "sass/style.scss",
                tasks: ["sass", "cssmin", "imagemin"]
            },
            html: {
                files: "**/*.html",
            },
            img: {
                files: "dist/images",
                tasks: ["imagemin"]
            }

        }
    });
    grunt.loadNpmTasks("grunt-contrib-sass");
    grunt.loadNpmTasks("grunt-contrib-watch");
    grunt.loadNpmTasks("grunt-contrib-cssmin");
    grunt.loadNpmTasks('grunt-contrib-imagemin');
    grunt.registerTask("default", ["watch"]);
}
```

terminal (https://gruntjs.com/plugins

1. **npm init**
    - ○ **(package.json)**
2. **npm install grunt -g**
    - ○ **(global)**
3. **npm install grunt –save-dev**
4. **npm install -g grunt-cli**
    - ○ **(ability to use grunt and global)**
5. **npm install grunt –save-dev**
    - ○ **(add grunt to the project)**
6. **npm install grunt-contrib-sass –save-dev**
    - ○ **(ability to process sass from scss fukes to css files)**
7. **npm install grunt-contrib-watch –save-dev**
    - ○ **(It watches and will alert if there is any error on the sass files)**
8. **grunt/stop grunt**
9. **npm install grunt-contrib-cssmin –save-dev**
    - ○ **(install cssmin)**
10. **npm install grunt-browser-sync –save-dev**
11. **grunt**
12. **pass/fail**