

Construcción de un proyecto AngularJS utilizando Yeoman.

En la presente guía se construirá una aplicación web SPA en AngularJS utilizando un software llamado Yeoman, el cual consiste en generar un scaffold de proyecto web (Estructura de aplicación con integración de plugins base para un espacio de trabajo).

En la anterior guía se trataron temas de instalación del entorno para trabajar y poder generar proyectos ahora daremos las instrucciones para generar el proyecto usando Yeoman.

1- Debemos crear un directorio, en este ejemplo lo llamaremos “ngproject”:

```
mkdir ngproject
```

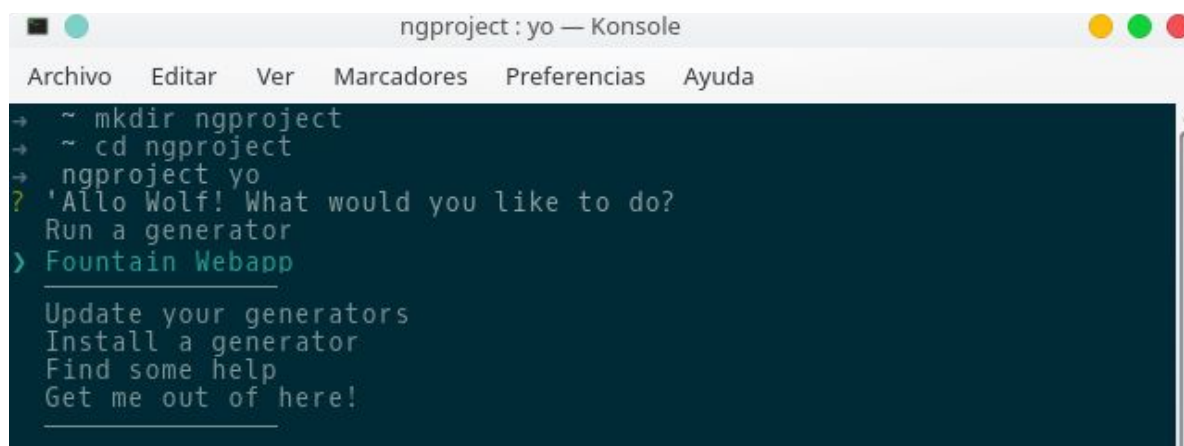
2- Nos posicionamos en el directorio creado.

```
cd ngproject
```

3- Ejecutamos el comando de Yeoman dentro del directorio donde construiremos la app.

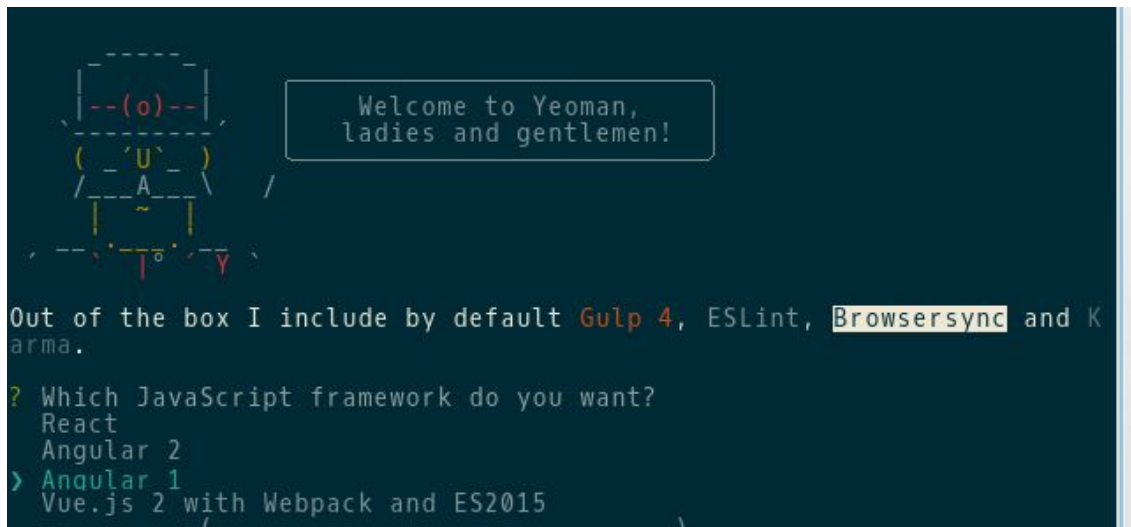
```
yo
```

4- Seleccionamos con el cursor la opción de “Fountain Webapp”, los pasos mencionados anteriormente están descritos en la siguiente imagen:



```
ngproject : yo — Konsole
Archivo  Editar  Ver  Marcadores  Preferencias  Ayuda
→ ~ mkdir ngproject
→ ~ cd ngproject
→ ngproject yo
? 'Allo Wolf! What would you like to do?
  Run a generator
> Fountain Webapp
  Update your generators
  Install a generator
  Find some help
  Get me out of here!
```

5- Seleccionamos con el cursor el framework de javascript, la opción es “Angular 1”



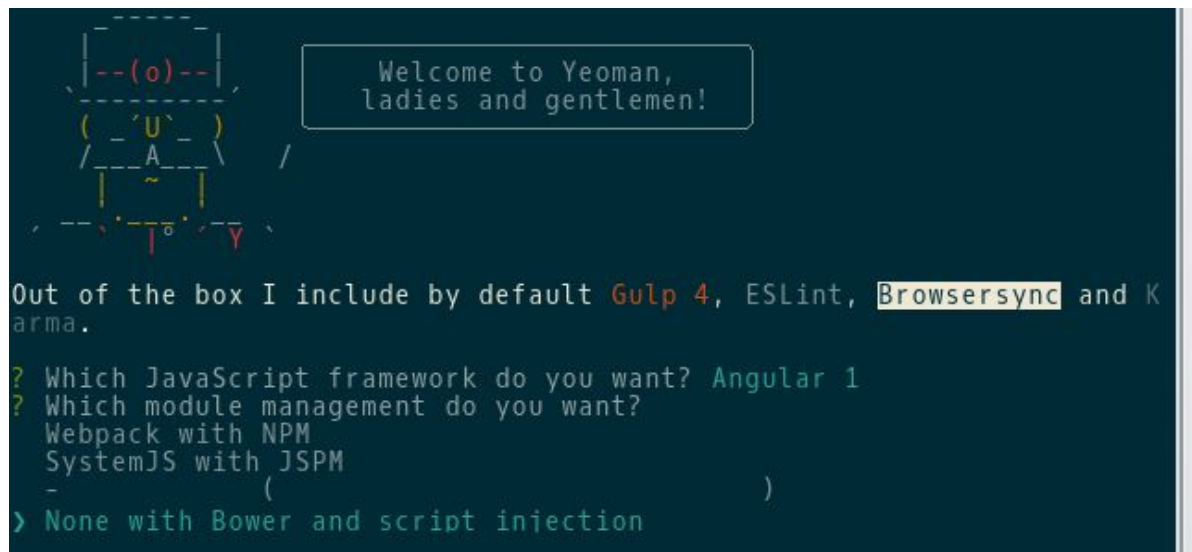
```
--(o)--
  _U_
 /  A  \
 |  ~  |
 |__|__|
 |  o  |
 |__|__|

Welcome to Yeoman,
ladies and gentlemen!

Out of the box I include by default Gulp 4, ESLint, Browsersync and Karma.

? Which JavaScript framework do you want?
  React
  Angular 2
> Angular 1
  Vue.js 2 with Webpack and ES2015
```

6- Después seleccionamos con el cursor el módulo de administración y seleccionamos la opción “None with Bower and script injection”:



```
--(o)--
  _U_
 /  A  \
 |  ~  |
 |__|__|
 |  o  |
 |__|__|

Welcome to Yeoman,
ladies and gentlemen!

Out of the box I include by default Gulp 4, ESLint, Browsersync and Karma.

? Which JavaScript framework do you want? Angular 1
? Which module management do you want?
  Webpack with NPM
  SystemJS with JSPM
  -
> None with Bower and script injection
```

7- Ahora deberemos seleccionar el “preprocessor” de JavaScript y seleccionamos “Pure old JavaScript” y damos enter.

Welcome to Yeoman,
ladies and gentlemen!

Out of the box I include by default Gulp 4, ESLint, Browsersync and Karma.

? Which JavaScript framework do you want? Angular 1
? Which module management do you want? None with Bower and script injection
? Which JS preprocessor do you want?
 ES2015 today with Babel
> Pure old JavaScript
 TypeScript

8- Como lo hicimos con JavaScript también debemos elegir el preprocesador de hojas de estilo, en este caso seleccionaremos “CSS”

ngproject:yo — Konsole

Archivo Editor Ver Marcadores Preferencias Ayuda

```

→ ~ mkdir ngproject
→ ~ cd ngproject
→ ngproject yo
? 'Allo Wolf! What would you like to do? Fountain Webapp

Make sure you are in the directory you want to scaffold into.
This generator can also be run with: yo fountain-webapp

  --(o)--
 /  _  \
(  _U_  )
/  _A_  \
 |  ~   |
 |  _   |
 \  _   /
  --(o)--

Welcome to Yeoman,
ladies and gentlemen!

Out of the box I include by default Gulp 4, ESLint, Browsersync and Karma.

? Which JavaScript framework do you want? Angular 1
? Which module management do you want? None with Bower and script injection
? Which JS preprocessor do you want? Pure old JavaScript
? Which CSS preprocessor do you want?
  SASS
  Stylus
  Less
> CSS
-
  )

```

9- En el siguiente paso nos pregunta si deseamos agregar un software para usar integración continua en nuestro proyecto, como no utilizaremos integración continua **SÓLO** damos en Enter.

```
Out of the box I include by default Gulp 4, ESLint, Browsersync and Karma.

? Which JavaScript framework do you want? Angular 1
? Which module management do you want? None with Bower and script injection
? Which JS preprocessor do you want? Pure old JavaScript
? Which CSS preprocessor do you want? CSS
? Which Continuous Integration platform do you want? (Press <space> to select, <a> to toggle all, <i> to inverse selection)
> ☐ Travis
  ☐ CircleCi
  ☐ Jenkins (with Dockerfile)
  ☐ Wercker
```

10- Ahora debemos seleccionar cual es la aplicación de ejemplo que elegiremos, en este caso seleccionaremos "A working landing page"

```
Out of the box I include by default Gulp 4, ESLint, Browsersync and Karma.

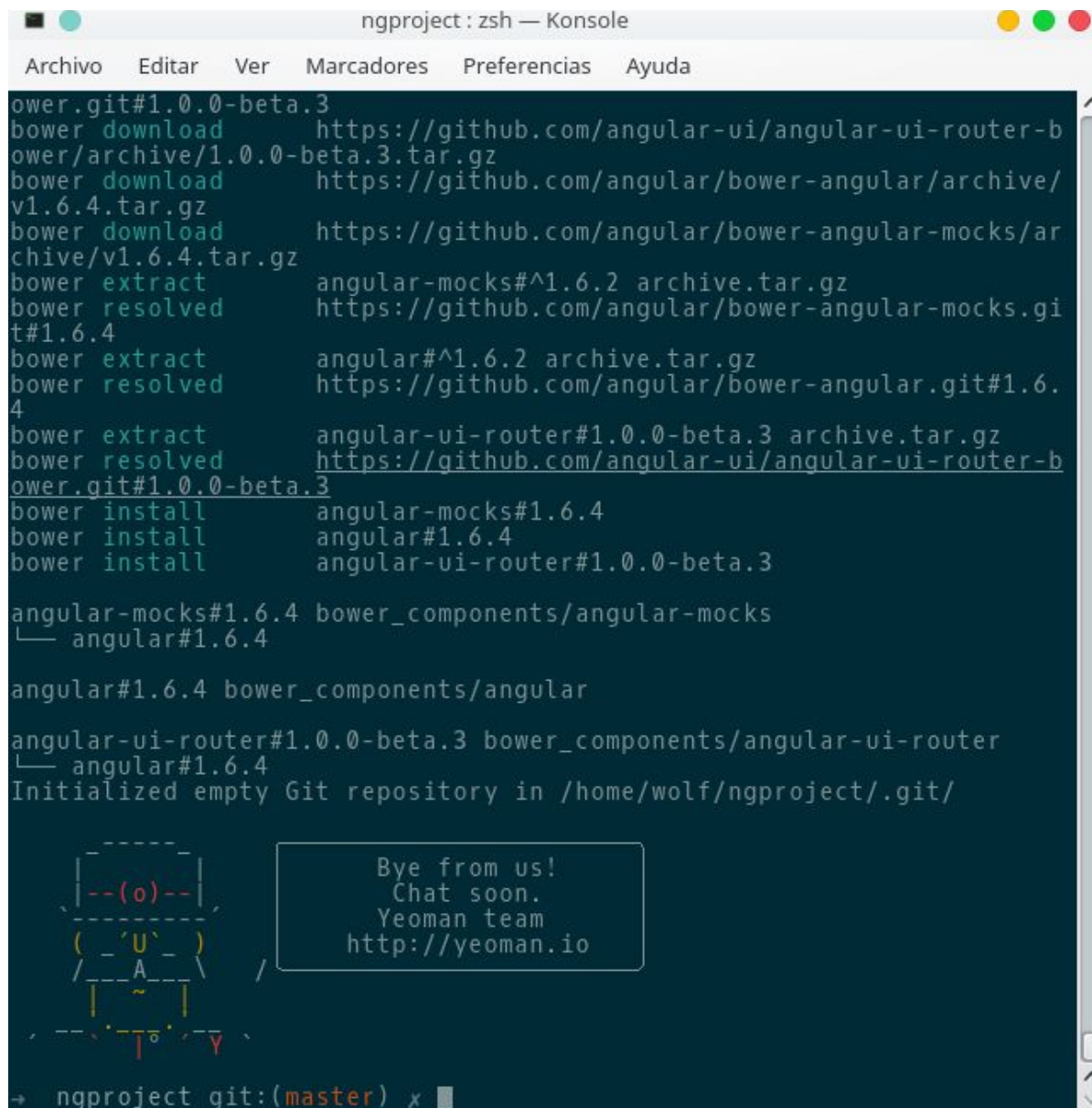
? Which JavaScript framework do you want? Angular 1
? Which module management do you want? None with Bower and script injection
? Which JS preprocessor do you want? Pure old JavaScript
? Which CSS preprocessor do you want? CSS
? Which Continuous Integration platform do you want? (Press <space> to select, <a> to toggle all, <i> to inverse selection)
? Do you want a sample app?
> A working landing page
  Just a Hello World
  TodoMVC
```

11- Para ir finalizando el proceso debemos seleccionar el router que usará la aplicación, en este caso solo podemos elegir "Angular UI Router"

```
Out of the box I include by default Gulp 4, ESLint, Browsersync and Karma.

? Which JavaScript framework do you want? Angular 1
? Which module management do you want? None with Bower and script injection
? Which JS preprocessor do you want? Pure old JavaScript
? Which CSS preprocessor do you want? CSS
? Which Continuous Integration platform do you want? (Press <space> to select, <a> to toggle all, <i> to inverse selection)
? Do you want a sample app? A working landing page
? Would you like a router? (Use arrow keys)
> Angular UI Router
  None
```


12- Finalmente tendremos este mensaje en nuestra terminal.



```
ngproject: zsh — Konsole
Archivo  Editar  Ver  Marcadores  Preferencias  Ayuda

ower.git#1.0.0-beta.3
bower download https://github.com/angular-ui/angular-ui-router-b
ower/archive/1.0.0-beta.3.tar.gz
bower download https://github.com/angular/bower-angular/archive/
v1.6.4.tar.gz
bower download https://github.com/angular/bower-angular-mocks/ar
chive/v1.6.4.tar.gz
bower extract angular-mocks#^1.6.2 archive.tar.gz
bower resolved https://github.com/angular/bower-angular-mocks.gi
t#1.6.4
bower extract angular#^1.6.2 archive.tar.gz
bower resolved https://github.com/angular/bower-angular.git#1.6.
4
bower extract angular-ui-router#1.0.0-beta.3 archive.tar.gz
bower resolved https://github.com/angular-ui/angular-ui-router-b
ower.git#1.0.0-beta.3
bower install angular-mocks#1.6.4
bower install angular#1.6.4
bower install angular-ui-router#1.0.0-beta.3

angular-mocks#1.6.4 bower_components/angular-mocks
└─ angular#1.6.4

angular#1.6.4 bower_components/angular

angular-ui-router#1.0.0-beta.3 bower_components/angular-ui-router
└─ angular#1.6.4

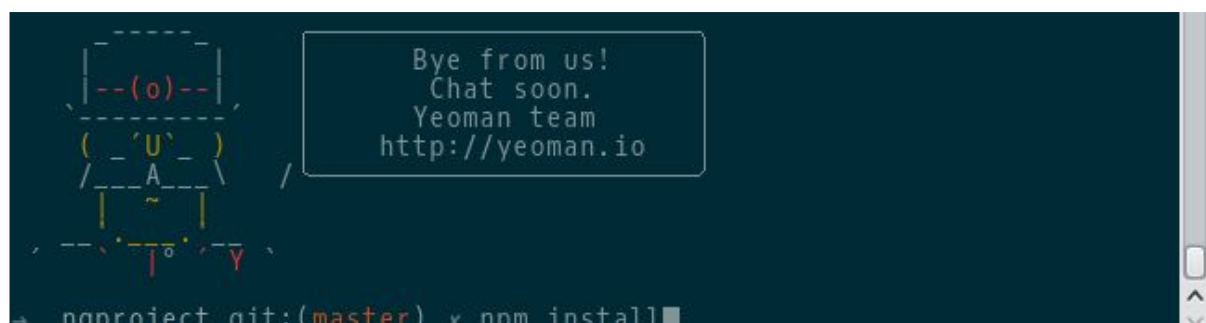
Initialized empty Git repository in /home/wolf/ngproject/.git/

  --(o)--
  (  'U'  )
  /  _A_  \
  |  ~    |
  |__T__Y_|

Bye from us!
Chat soon.
Yeoman team
http://yeoman.io

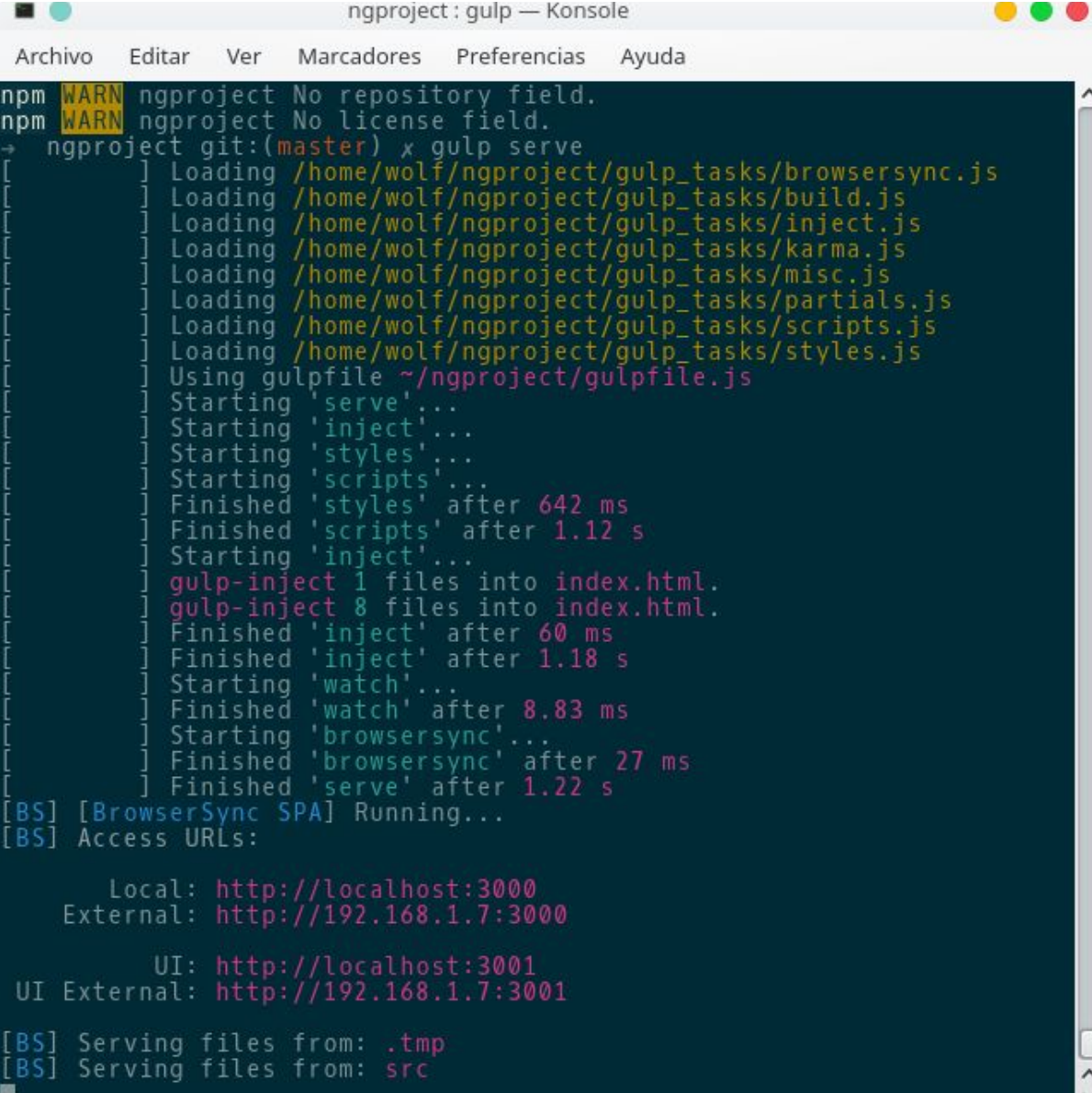
→ ngproject git:(master) x
```

13- Ahora Ejecutaremos “**npm install**” para terminar de instalar las dependencias de nuestra aplicación.



```
ngproject git:(master) x npm install
```

14- Para ejecutar nuestro proyecto deberemos ejecutar “gulp serve” en nuestra terminal y obtendremos la siguiente captura.

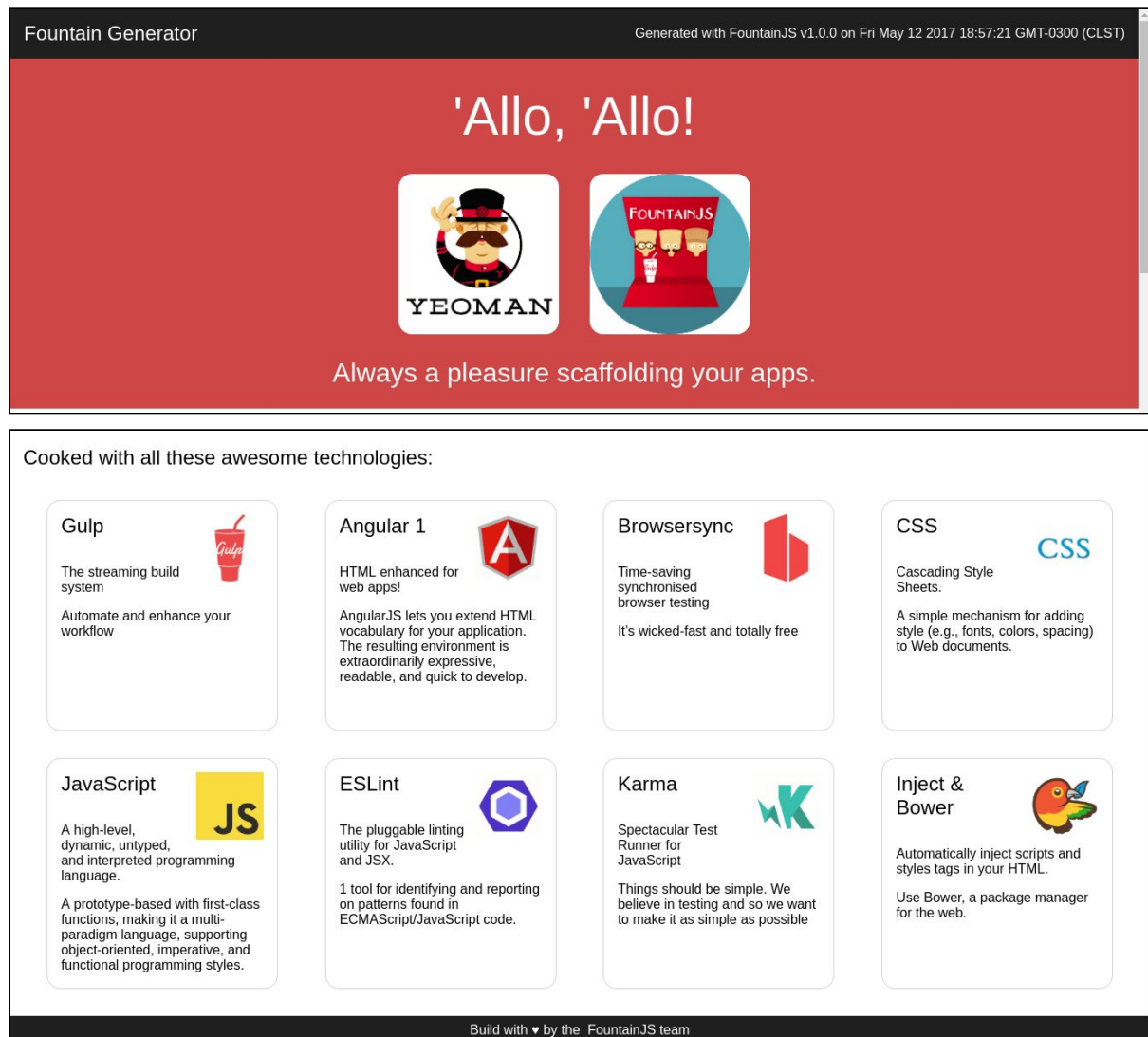


```
ngproject : gulp — Konsole
Archivo  Editar  Ver  Marcadores  Preferencias  Ayuda

npm WARN ngproject No repository field.
npm WARN ngproject No license field.
→ ngproject git:(master) x gulp serve
[ ] Loading /home/wolf/ngproject/gulp_tasks/browsersync.js
[ ] Loading /home/wolf/ngproject/gulp_tasks/build.js
[ ] Loading /home/wolf/ngproject/gulp_tasks/inject.js
[ ] Loading /home/wolf/ngproject/gulp_tasks/karma.js
[ ] Loading /home/wolf/ngproject/gulp_tasks/misc.js
[ ] Loading /home/wolf/ngproject/gulp_tasks/partials.js
[ ] Loading /home/wolf/ngproject/gulp_tasks/scripts.js
[ ] Loading /home/wolf/ngproject/gulp_tasks/styles.js
[ ] Using gulpfile ~/ngproject/gulpfile.js
[ ] Starting 'serve'...
[ ] Starting 'inject'...
[ ] Starting 'styles'...
[ ] Starting 'scripts'...
[ ] Finished 'styles' after 642 ms
[ ] Finished 'scripts' after 1.12 s
[ ] Starting 'inject'...
[ ] gulp-inject 1 files into index.html.
[ ] gulp-inject 8 files into index.html.
[ ] Finished 'inject' after 60 ms
[ ] Finished 'inject' after 1.18 s
[ ] Starting 'watch'...
[ ] Finished 'watch' after 8.83 ms
[ ] Starting 'browsersync'...
[ ] Finished 'browsersync' after 27 ms
[ ] Finished 'serve' after 1.22 s
[BS] [BrowserSync SPA] Running...
[BS] Access URLs:
      Local: http://localhost:3000
      External: http://192.168.1.7:3000
      UI: http://localhost:3001
      UI External: http://192.168.1.7:3001
[BS] Serving files from: .tmp
[BS] Serving files from: src
```

“gulp serve” es un comando definido en nuestro “package.json” y además en nuestros archivos de gulp, este contiene todas las tareas necesarias para que gulp construya la carpeta “tmp” en nuestro proyecto y esta tiene el livereload de nuestro proyecto, es una copia del proyecto donde se puede apreciar la inyección de scripts.

15- Después de que Gulp.JS compile y ejecute todos nuestros códigos levantará la aplicación con un servidor Node y esta estará dispuesta en el puerto 3000:

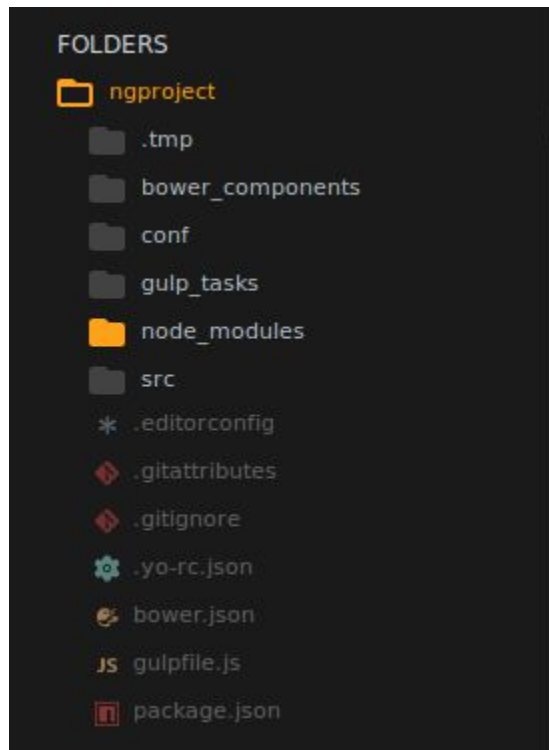


En esa imagen se muestra las tecnologías que integra todo nuestro proyecto de AngularJS, donde Gulp es nuestra herramienta de automatización de tareas para FrontEnd, AngularJS es nuestro FrameWork SPA de JavaScript, Browsersync es el módulo que nos permite integrar pruebas en nuestro navegador para ahorrar tiempo (LiveReload), Karma js es una biblioteca para hacer Test en JS y además integra inyección de Scripts mediante el uso de los plugins instalados por bower.

16- Finalmente, podemos revisar nuestro proyecto generado por Yeoman, para ello utilizaremos nuestro editor de texto Sublime Text.



17- Estructura de Proyecto:



El módulo principal donde estarán nuestros scripts JS está en “src/”, como lo muestra la siguiente imagen:



En “app/” tendremos todos nuestros archivos JS generados por Yeoman:

