Saliendo a Producción

Módulo 4 - Lección 3: Saliendo a producción

En esta lección trabajaremos sobre las tareas necesarias para salir a producción con nuestro proyecto. Veremos las prácticas habituales en la industria y luego haremos una breve práctica utilizando un proyecto de juguete donde utilizaremos las herramientas más utilizadas para estas tareas.

De acuerdo a los diferentes esquema de desarrollo de software, los pasos típicos que se siguen en todo proyecto son:

1. Relevamiento de las necesidades del proyecto
2. Análisis de los requerimientos
3. Diseño de la solución
4. Programación
5. Testing
6. Puesta en producción

Todos estos pasos se dan de forma iterativa e incremental. Iterativa porque se repiten cada cierto tiempo convenido, por ejemplo cada 2 semanas, e incremental porque se van integrando los desarrollos de la solución.
Dependiendo del autor y la matedología, pueden cambiar los pasos incluyendo análisis de riesgos, colaboración del cliente, categorización de features, etc, pero podemos pensarlas como subetapas contenidas dentro de alguno de los pasos previos.

Enunciado este marco de trabajo, nosotros estamos en el último paso: la puesta en producción.

¿En qué consiste?

En términos generales, consiste en publicar nuestro projecto al cliente en un ambiente productivo, es decir, el ambiente que utilizará el cliente para trabajar.
En el caso de nuestro proyecto nodejs con Bootstrap, lo que haremos es publicarlo en un servidor web para que, cualquier persona, que queramos pueda acceder al mismo.

Nota que como paso previo se incluye el testing. El testing se entiende a la actividad de validar y verificar nuestra solución. La validaremos con el cliente, asegurandonos que desarrollamos el producto correcto y verificaremos la solución, asegurandonos que la desarrollamos correctamente.
Para la validación, se suele trabajar con demostraciones con el cliente donde le mostraremos frecuentemente en un plazo convenido (2 o 3 semanas habitualmente), los avances de nuestro proyecto permitiendo realizar cambios a tiempo. En el trabajo con el equipo es importante tener bien en claro la definición de *tarea terminada.* Te comento esto en este lugar porque cuando estés con varias personas en un mismo proyecto verás que cada uno tiene su propia definición de tarea terminada y es importante acordar los términos. Algunos pueden pensar que tarea terminada no incluye el testing, otros que tarea terminada sólo aplica cuando el

cliente la aprueba, otros cuando incluye la seguridad y testing, etc. Lo recomendable es escribir lo que se conoce como *test de aceptación* en cada tarea. Esto se define con el cliente y se establecen los pasos con los criterios de aceptación para cada tarea. Asimismo, es importate contar con los tests de performance (si aplica), unitarios, de integración, de seguridad. Quizá te parezca demasiado todo esto y escapa al alcance del curso, pero son términos que veras en tu día a día de trabajo y es bueno que los vayas explorando por tu cuenta.
Para esta etapa de testing con el cliente, muchos equipos crean un ambiente de testing, *staging* como se suele nombrar, o pre-producción. Este ambiente replica las condiciones del ambiente productivo pero sólo sirve para realizar pruebas y mostrar avances con el cliente. En proyectos grandes se suele tener más de uno y se agregan por etapas hasta llegar a los servidores productivos. Por lo tanto, la salida a producción en estos casos es pasar de pre-producción a producción, tarea que implica menores pasos que lo que veremos a continuación porque ya se han hecho antes.

¿Qué sucede con nuestro proyecto?

En primer lugar, en nuestro caso omitiremos el paso de testing dado que no tenemos un cliente más que las guías que vamos desarrollando, y los testing de javascript los haremos en los cursos posteriores. Haremos foco en las tareas de puesta en producción de todo lo demás. Las mismas consistirán en lo que respecta a CSS:

1. Compilar el código Sass o Less a CSS
2. Ejecutar Autoprefixer para agregar/eliminar los prefijos que requieran los diferentes browsers en nuestros archivos .css
3. Minification para eliminar todo espacio o carácter innecesario en nuestros archivos sin cambiar las funcionalidades desarrolladas y reducirlos al mínimo tamaño posible.
4. Concatenación de los archivos para generar un único css que agiliza la carga de nuestro sitio

En lo que respecta a JS:

1. Ejecutar JSHint, que es una herramienta de análisis estático de código que muestra potenciales problemas de javascript que serán útiles analizar antes de salir a producción
2. Concatenación de los archivos para generar un único js que agiliza la carga de nuestro (idem con css)
3. Uglification: minification + mangling. Esta tarea de nombre extraño implica minificar los archivos como explicamos anteriormente y agregarles la etapa de mangling que consiste en cambiar los nombres de las variables por letras o combinación de las mismas. Esto tiene 2 beneficios, por un lado reducir el tamaño de los archivos y por otro, dificultarle a un curioso el estudio de nuestros archivos js
4. Rechequeo que errores para asegurarnos que no se generó ningún error en la ejecución de los pasos previos

Para otros tareas:

1. Comprimir imágenes de nuestro sitio

2. Observar los cambios en los archivos y automáticamente volver a ejecutar las tareas antes mencionadas. Además de ahorrarnos tiempo, tendremos el proceso de ejecución asegurado automáticamente
3. Configurar un server activo y sincronizado con nuestro código, cómo hemos visto previamente las librerías de liteserver y browserfy. Esto nos ahorrará mucho tiempo y sobretodo, nos permitirá tener foco en el desarrollo
4. Testear
5. Generar los archivos para salir a producción, que consiste en crear los *distribution files* o archivos de distribución que son los que alojaremos en el servidor productivo para que todos puedan acceder desde la web

Todas estas tareas podemos hacerlas manualmente pero serían muy tediosas y propensas a errores involuntarios. Por ese motivo, es que existen automatizadores de tareas como Grunt y Gulp que veremos en el siguiente módulo. La filosofía dentras de este principio se lla DRY, acrónimo del inglés *don´t repeat yourself,* que podemos traducirla a *no hacer dos veces lo mismo.* Por lo tanto, usaremos automatizadores posteriormente.

El siguiente tutorial, trabajaremos con estas herramientas para familiarizarnos con la gestión productiva de un proyecto.