Universidad Don Bosco

Docente:

Edwin Bonilla

Estudiantes:

Katherine Beltrán BL233081

Michael Hernández CH230228

Jonatan Cardoza CP230528

Diego Peña PV230210

Ismael Rivera RL233297

Andrea

Tema:

Mockups Definición y Uso y su Papel en el Desarrollo Web

Asignatura:

Lenguajes de Marcado y Estilo Web

Grupo:

LME404 G01T

Fecha de Entrega:

11 de septiembre de 2023

Introducción

La mayoría de desarrolladores por decir en realidad todos utilizarán algún tipo de sistema de control de versiones (VCS), una herramienta que les permita colaborar con otros desarrolladores en un proyecto sin peligro de que sobrescriban el trabajo de los demás, y volver a las versiones anteriores de la base de código si existe un problema descubierto más tarde.

El más popular (al menos entre los desarrolladores web) es Git, junto con GitHub, un sitio que proporciona alojamiento para tus repositorios y varias herramientas para trabajar con ellos. Este módulo tiene como objetivo enseñarte lo que necesitas saber sobre ambos.

En el siguiente proyecto veremos la definición y uso de los Mockups y la importancia de estos en el trabajo y actividades diarias en el uso de desarrollo web, mencionando como algunos de estos nos ayudan a linear objetivos y expectativas personales del proyecto, pero sobre todo del cliente.

También veremos algunas herramientas más populares para crear Mockups y los ejemplos como podemos utilizar estas herramientas para el desarrollo de nuestra página web, mostrando el proceso de creación del Mockup para nuestra página web y representando con elementos visuales las interacciones con este.

También aprenderemos acerca de "commits" y "pull requests" en GitHub la definición de estos y el uso que le daremos en el desarrollo de nuestra página web.

Y mostramos el desarrollo de nuestra página web utilizando estas grandiosas herramientas.

Mockups Definición y Uso y su Papel en el Desarrollo Web

Mockups

Un mock-up (o mockup) es un modelo que se utiliza para representar de forma rápida el resultado final de un diseño, es decir que es una herramienta indispensable en nuestro día a día y qué se usa en el mundo del diseño gráfico.

Se trata de una palabra de origen inglés que proviene del verbo compuesto mock up, que significa bosquejar o bosquejo, en definitiva, es un montaje que simula el resultado de un producto y que sirve para obtener la aprobación del cliente o consumidor final, en pocas palabras es el canal a través del cual podemos presentar un proyecto y demostrar cómo quedará, tanto en lo estético como en lo funcional.

Los mockups son importantes en el proceso de desarrollo web y propósito principal de un mockup es el de validar nuestras ideas antes de mandarlas a impresión o tomar decisiones importantes, y de esta manera permitirnos presentar nuestro trabajo de forma profesional, pero ahorrándonos los costes de impresión y de montaje, y abriendo paso a ser muy útiles para que el cliente o consumidor de ese producto y pueda indicar qué cambios aplicaría o qué desearía mejorar de ese proyecto.

<u>Existen diferentes tipos de mockups:</u>

Aunque hayamos hablado del Mockup en la fase de creación de una web, debemos saber que existen distintos tipos de Mockup en función de cuál sea el producto final al que corresponden, es decir cada tipo de Mockup tiene sus características y no hay dos Mockups iguales. Por ejemplo, si tu necesitas una nueva página web, es de preparar el Mockup de la

misma a partir del briefing de ideas que se haya determinado con anterioridad, esto para poder hacernos una idea clara de lo que será tu página web una vez creada, considerando que es indispensable que este Mockup muestre claramente cuáles serán los menús, las funcionalidades y los apartados de dicha web.

Estos son algunos de los tipos de Mockup que existen, pero lo cierto es que pueden ser infinitos:

- Mock Up para productos de merchandising
- Mock Up para página web
- Mock Up para blog
- Mock Up para folletos y flyers
- Mock Up para productos de cartelería

En las diferentes disciplinas del diseño podemos ver que el mockup sirve para lo mismo. A primera vista, encontramos estos 4 usos que nos permiten entender mejor qué es un mockup:

Ayuda a ver la realidad

Este puede resultar el más evidente. No todos percibimos las ideas de la misma forma, por lo que un mockup nos ayudará a situarnos en el mismo punto que nuestro cliente y compañeros de equipo, viendo de la manera más fiel posible los resultados que queremos obtener de nuestro proyecto.

Acelera las decisiones a tomar en el proyecto

De manera ordenada, podemos acelerar las decisiones a tomar en el desarrollo del proyecto utilizando los mockups.

Y es mucho más sencillo discutir una idea y qué dirección debemos tomar si tenemos un resultado aproximado de cómo va a quedar en lugar de hablar todo sin ningún refuerzo visual, y de esta manera, podemos hacernos entender mejor y optimizar así la comunicación.

Ahorrar tiempo

Gracias a usar mockups, podemos disminuir el tiempo que invertimos en las correcciones de nuestro proyecto.

En definitiva, el mockup es un medio que nos permite plantear rápidamente el resultado de diferentes ideas, visualizando con mayor velocidad lo que funciona y lo que no dentro de un proyecto.

Ahorrar gastos

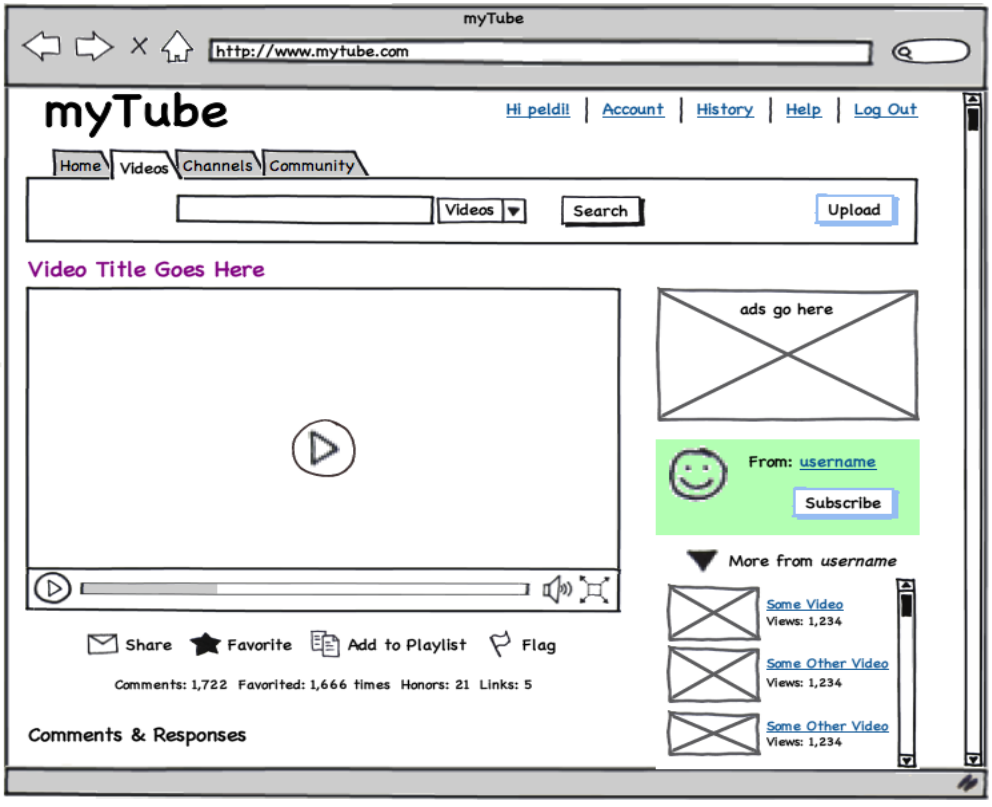
Cuando realizamos mockups-también conocidos como ficticios en diseño gráfico- durante el proyecto, disminuimos el riesgo de que el resultado final no agrade al cliente.

El uso del mockup permite un avance en el que la visión del cliente está en armonía con el diseño y desarrollo del proyecto, ahorrándonos correcciones tras la entrega del mismo.

Herramientas para Crear Mockups

Hay diferentes maneras de acceder a nuevas herramientas con las que mejorar el proceso de crear un prototipo que merezca la pena, aparte de que no todo el mundo es capaz de visualizar un proyecto para darle finalmente la forma definida.

Por esta razón existen diversas herramientas para crear mockups y wireframes como las algunas que nos ayudarán rápidamente a diseñar y probar vuestro nuevo proyecto.

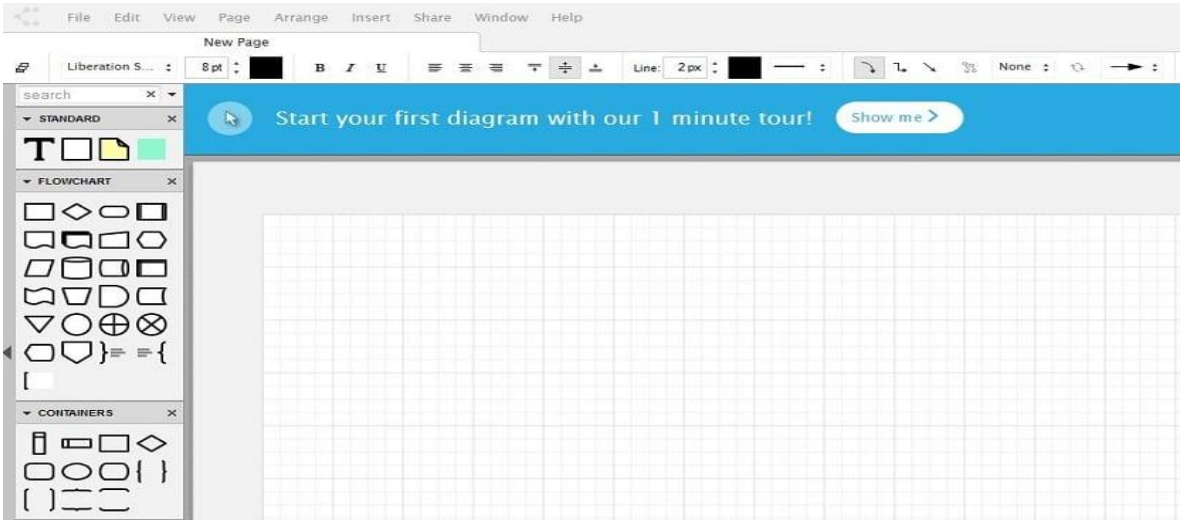


Balsamiq

Es una de las herramientas más importantes para crear tus primeros wireframes y mockups. Es fácil y sencilla de usar para tener un buen mockup preparado más rápido de lo que uno podría llegar a pensar a primeras. Si no se necesita interacciones con el prototipo es la elección perfecta.

Lucidchart

Se caracteriza por una interfaz que opera al arrastrar y soltar los elementos para crear el mockup. Se puede compartir el proyecto con otros y en un poco de tiempo se puede pasar de tener un proyecto rápido a uno que sería el producto final.

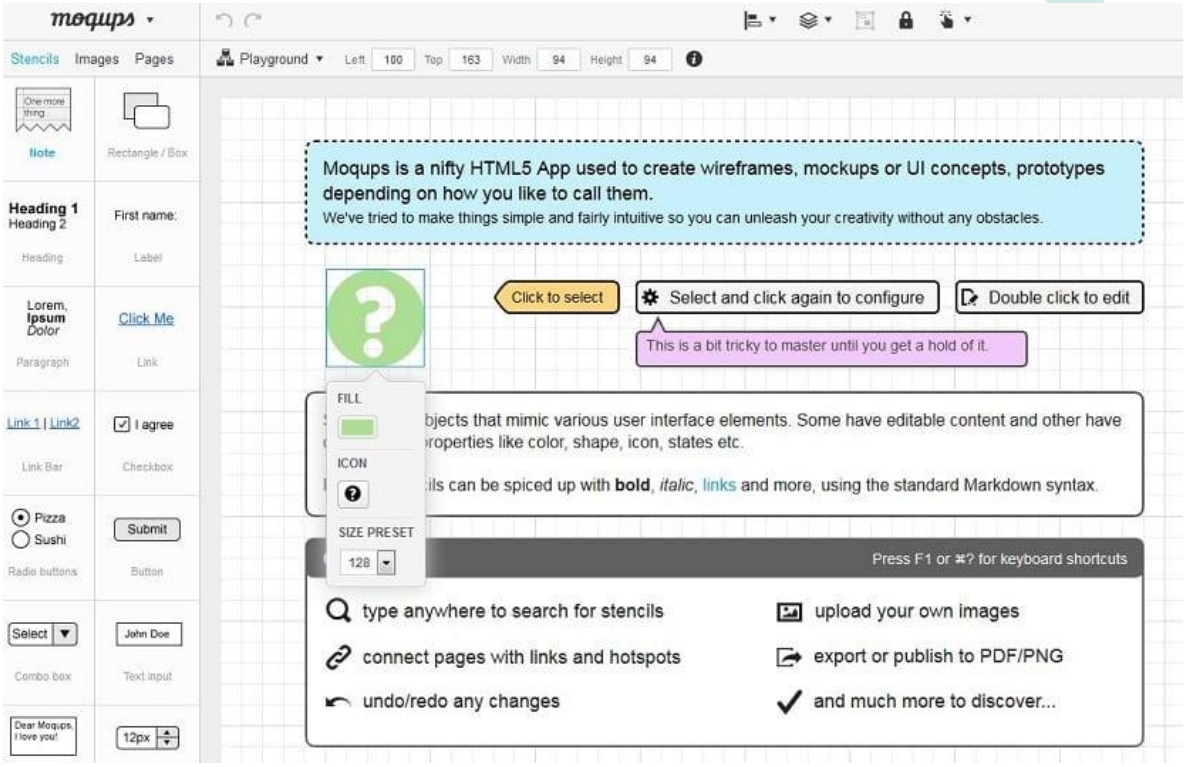


MarvelApp

Una de sus virtudes es la accesibilidad a través de distintas plataformas como puede ser PC, móvil o web. Se sincroniza con Dropbox y ofrece acceso a equipos completos. Otra de sus características distintivas es que se puede exportar el prototipo como si fuera un apk para desarrolladores Android.

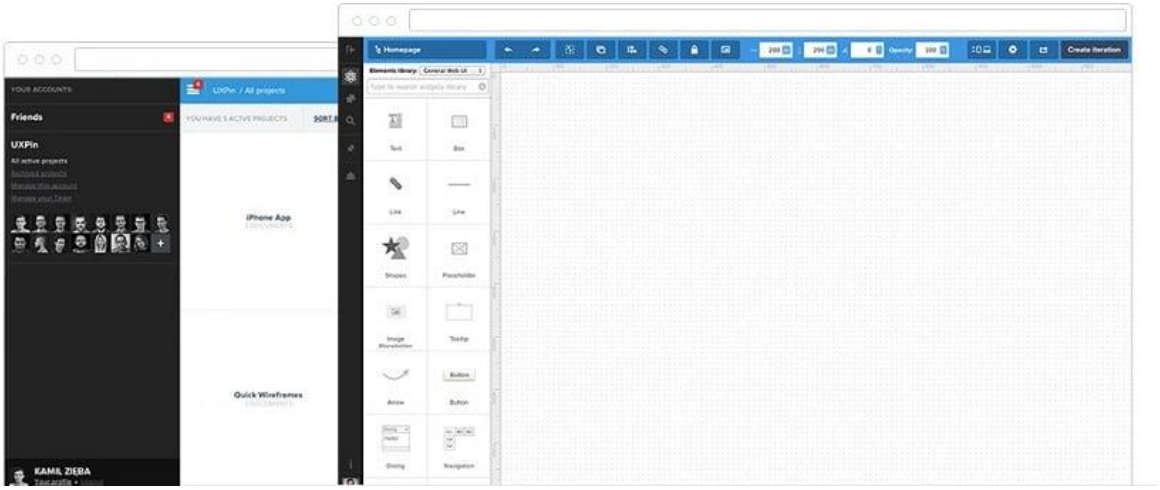
Moqups

Moqups es rápido y fácil al igual que Balsamiq. El hecho de que esté bajo HTML5 lo convierte en una herramienta rápida y sencilla. A destacar que la cantidad de miembros en el proyecto es ilimitada.



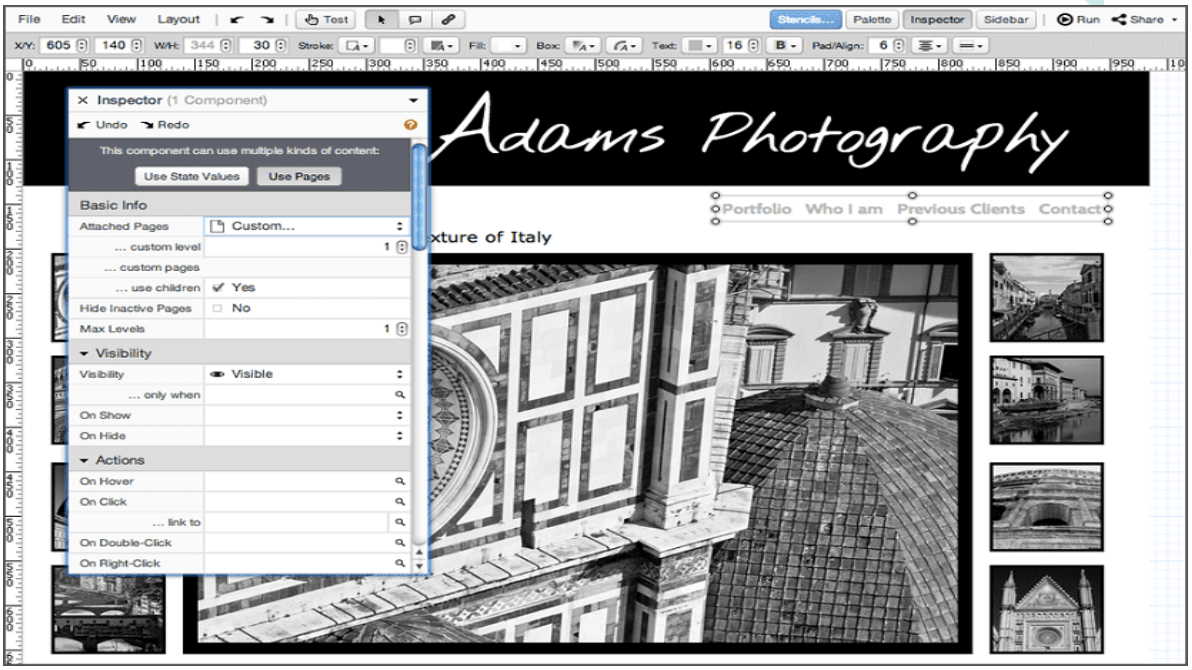
UXPin

Una gran herramienta con un montón de opciones para wireframing y que entre las cuales destaca la posibilidad de convertir un boceto en un wireframe digital. Y no es solo tomar un pantallazo y ya está, sino que conlleva todo lo necesario para convertir una imagen en digital. No es precisamente una herramienta barata, pero es de gran ayuda.



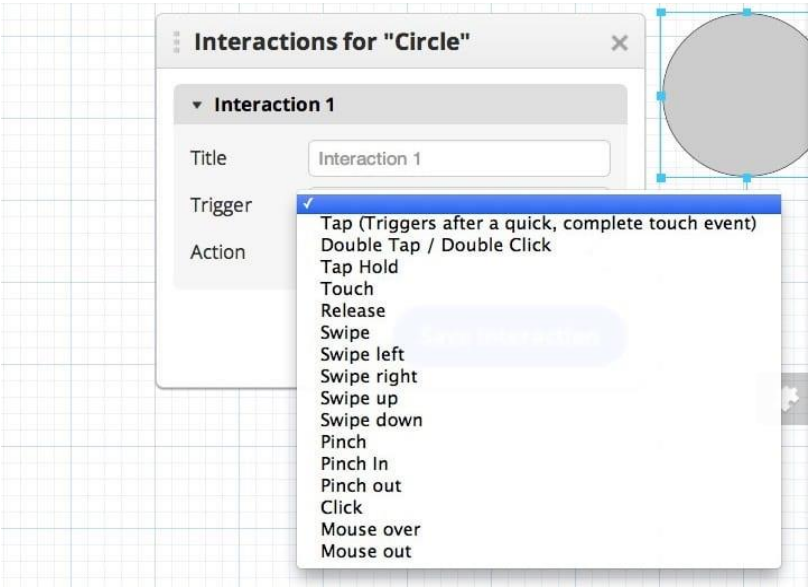
Protoshare

La virtud innata de Protoshare es su capacidad para el trabajo en equipo. Una herramienta profesional que permite discutir todos los proyectos en tiempo real con los colaboradores. Con una gran cantidad de opciones y al ser una herramienta bien clara en la interfaz le obliga a uno casi a tener que probarla.



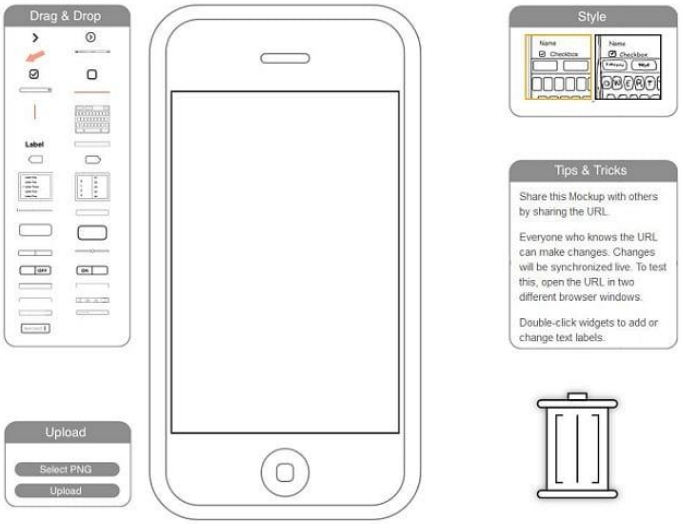
Proto.io

El sitio web de proto.io tiene un diseño bien simple y esto mismo es su gran valía. Una herramienta de gran valor para el prototipado con numerosas opciones para iOS, Android y Windows Phone.



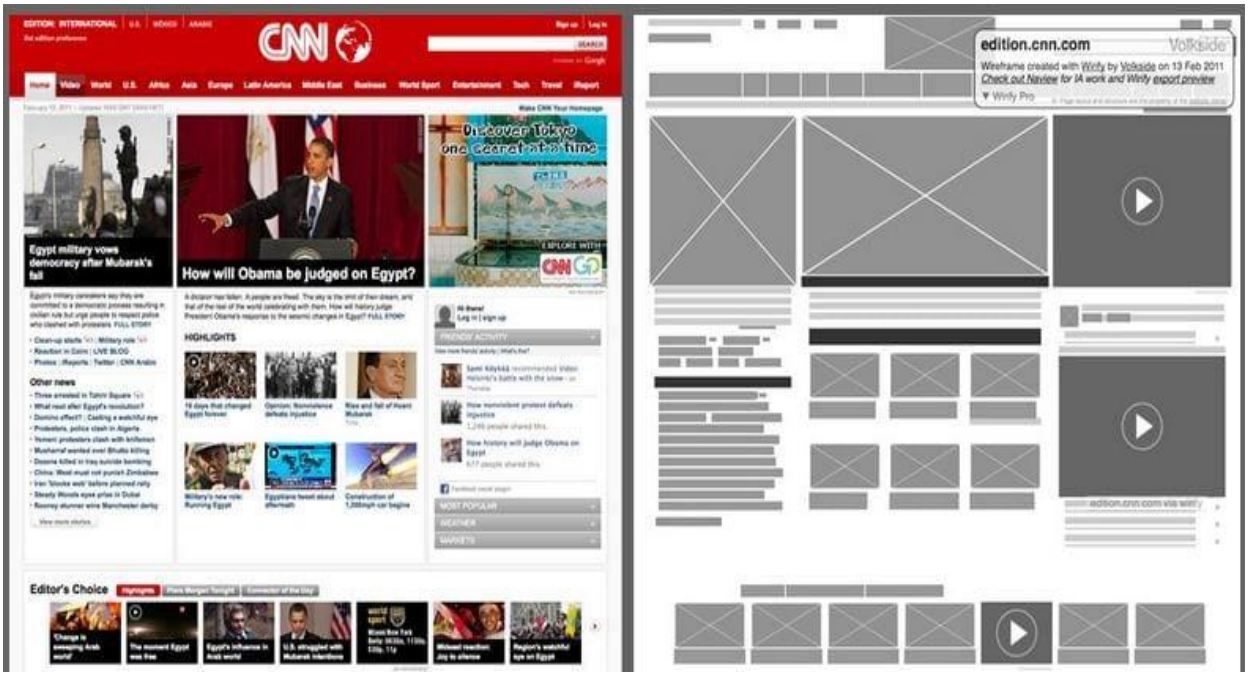
iPhoneMockup

La herramienta perfecta para los que no quieran sufrir mucho al pasar una imagen del papel a lo digital. Se elige uno de los estilos y se usa algunos de los elementos para crear un simple mockup para iOS.



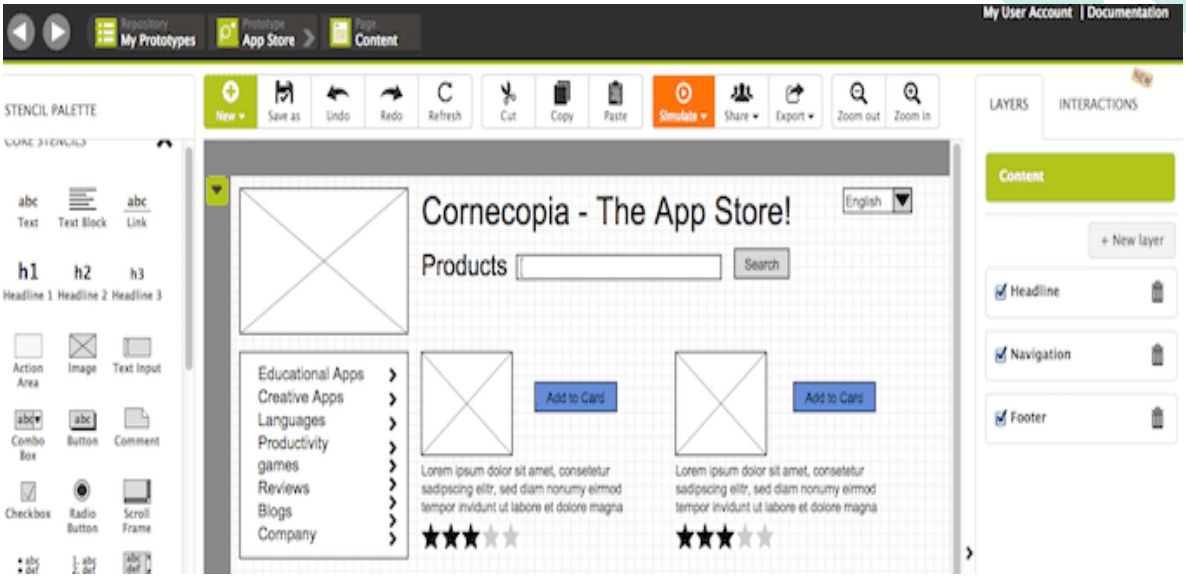
Wirify

Wirify da un script que convierte un sitio web en un wireframe. Es de gran ayuda cuando uno mismo encuentra una web que le llama mucho la atención y quiere usarla como base para su nuevo proyecto.



Pidoco

Se puede usar esta herramienta para probar los prototipos al enviar un enlace a los usuarios y así empezar a tomar datos de los mismos.



Creación y Diseño de Mockups

Actualmente, la red ofrece miles de opciones para descargar Mockups ya creados que nos pueden ayudar en el proceso de aprobación de algunos proyectos de diseño. Pero, siempre se recomienda que utilices tus propios Mockups, ya que sufrimos el riesgo de que nuestro mockup lo tenga todo el mundo y nos cueste más destacar.

La forma más fácil es descargar un mockup editable en alguna de las múltiples páginas que los ofrecen, tales como Freepik o Behance. Estos archivos pueden editarse con programas como Photoshop, ya que han sido preparados para que podamos sustituir la imagen que se muestra por nuestro diseño.

Al momento de crear un Mockup es importante seguir estos pasos:

A. Crea la arquitectura de la página en base al contenido del proyecto: categorías, páginas y menús.
B. Establece los elementos comunes de todas las páginas: header, footer y barras laterales.
C. Define el diseño de cada una de las páginas incluidas en la web.
D. Construye las maquetas de cada página con su contenido digital.

Procura pensar en qué necesitará el usuario o consumidor que visite la página web y procura ponérselo fácil en todos los sentidos.

GitHub y su Papel en el Desarrollo Web

Si te dedicas al software de control de versiones, GitHub es uno de los primeros programas con los que toparas. Pero ¿qué es GitHub exactamente? Desde 2008, la aplicación web ha ofrecido a los usuarios de todo el mundo la posibilidad de gestionar los procesos de desarrollo de sistemas y aplicaciones de software.

Github es un repositorio online gratuito que permite gestionar proyectos y controlar versiones de código. Es muy utilizado por desarrolladores para almacenar sus trabajos dando así la oportunidad a millones de personas de todo el mundo a cooperar en ellos.

Se podría hablar de Github como la red social pensada para desarrolladores, siendo este repositorio uno de los más usados a nivel mundial.

Más de 65 millones de usuarios aseguran que GitHub tiene casi el monopolio entre los distintos sistemas de control de versiones del mercado. En 2018, la empresa Microsoft compró GitHub por más de 7000 millones de dólares. Desde 2020, el servicio basado en la red también ofrece una aplicación para teléfonos inteligentes para los dos sistemas operativos más conocidos-iOS y Android-, que te permite acceder a tus proyectos y a su estado actual desde cualquier parte del mundo.

Podemos seguir e interactuar con personas interesadas en un tipo de proyecto en concreto, dando a conocer los nuestros o cooperando en el proyecto de terceros.

Control de Versiones con Git:

La palabra “Git” significa = Guía rápida para control de versiones, pero ¿Qué es un control de versiones?

Un control de versiones permite a los desarrolladores **administrar cambios en un software** a la vez que el proyecto evoluciona. En el caso de que un desarrollador quisiera trabajar en un proyecto, sería arriesgado realizar cambios sobre el código original. El control de versiones permite duplicar una parte de un proyecto de forma aislada y trabajar sobre ella **sin que se modifique el repositorio original**.

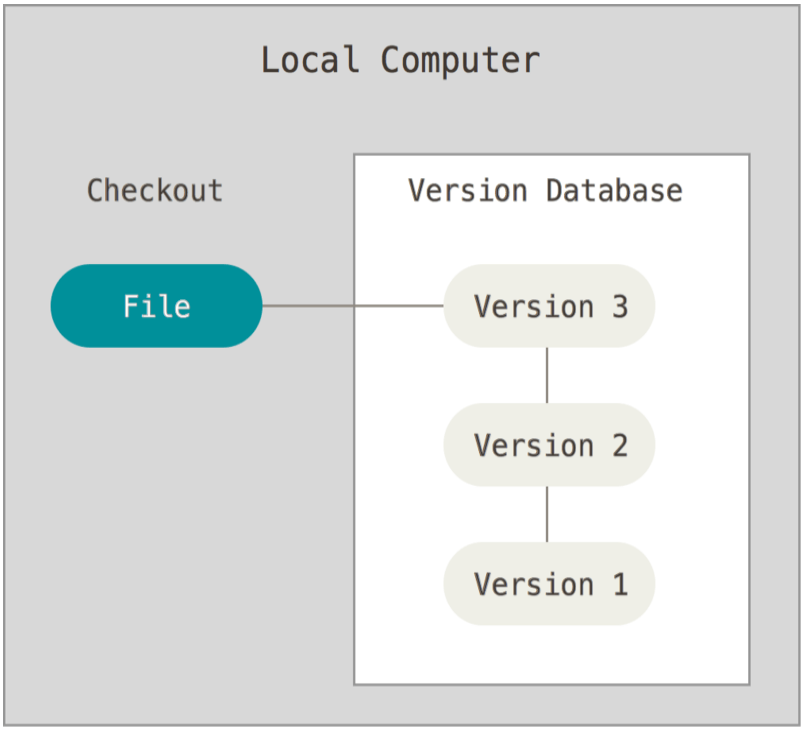
Una vez comprobado que el cambio se ha realizado con éxito, el desarrollador podrá fusionar su ramificación con el proyecto creando una nueva versión del mismo. Esta nueva versión registra los cambios realizados sobre la versión anterior para que se puedan testear por otros desarrolladores.

Cuando eres diseñador gráfico o de web y quieres mantener cada versión de una imagen o diseño, usar un sistema de control de versiones (VCS por sus siglas en inglés) es una decisión muy acertada. Usar un VCS también significa generalmente que si arruinas o pierdes archivos, será posible recuperarlos fácilmente.

Existen diferentes tipos de sistemas Git, y alguno de ellos son los siguientes:

Sistemas de Control de Versiones Locales:

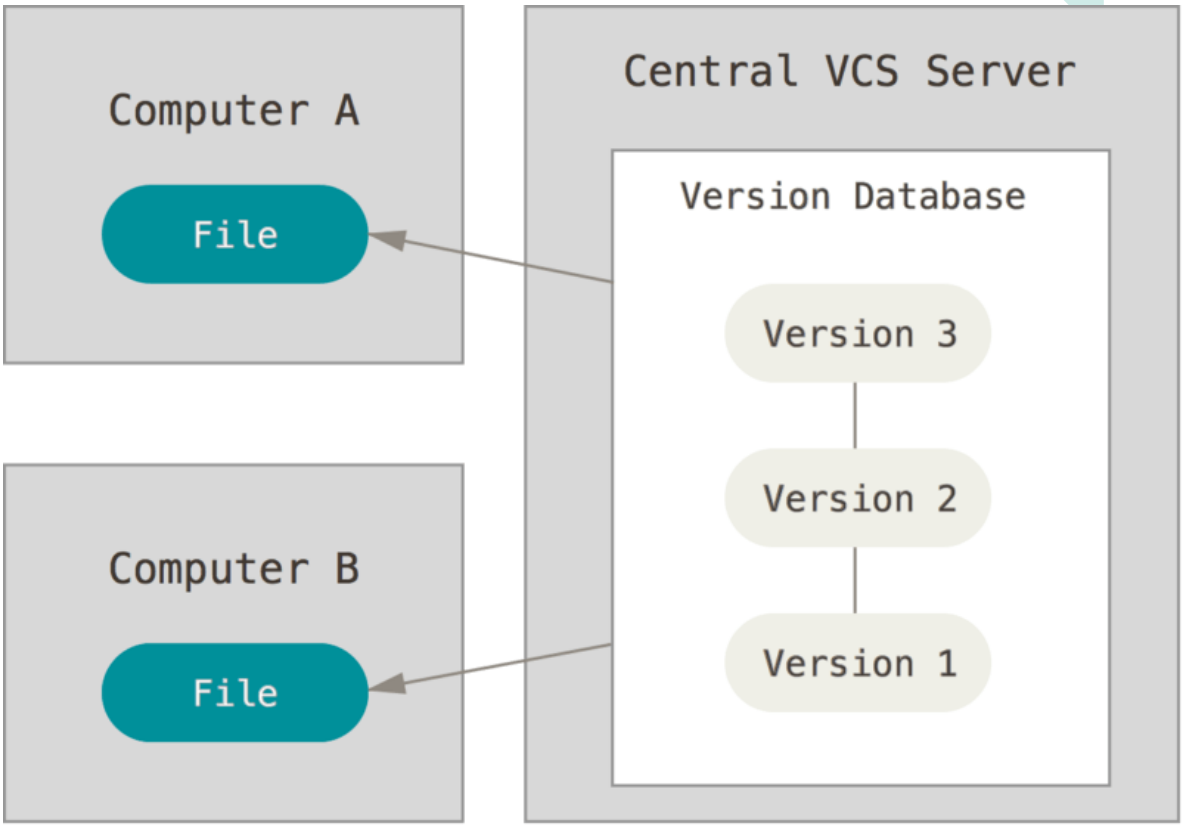
Usado por la mayoría, y consiste en copiar los archivos a otro directorio (quizás indicando la fecha y hora en que lo hicieron, si son ingeniosos). Este método es muy común porque es muy sencillo, pero también es tremendamente propenso a errores. Es fácil olvidar en qué directorio te encuentras y guardar accidentalmente en el archivo equivocado o sobrescribir archivos que no querías.



Para afrontar este problema los programadores desarrollaron hace tiempo VCS locales que contenían una simple base de datos, en la que se llevaba el registro de todos los cambios realizados a los archivos.

Una de las herramientas de control de versiones más popular fue un sistema llamado RCS, (Rich Communication Services), es un servicio de mensajería avanzada de Google. Se trata de un sistema para mandar SMS gratuitos sin necesidad de estar conectado a la red) que todavía podemos encontrar en muchas de las computadoras actuales.

Sistemas de Control de Versiones Centralizados: Los sistemas de Control de Versiones Centralizados (CVCS por sus siglas en inglés) fueron desarrollados para solucionar este problema. Estos sistemas, como CVS, Subversion y Perforce, tienen un único servidor que contiene todos los archivos versionados y varios clientes que descargan los archivos desde ese lugar central. Este ha sido el estándar para el control de versiones por muchos años.

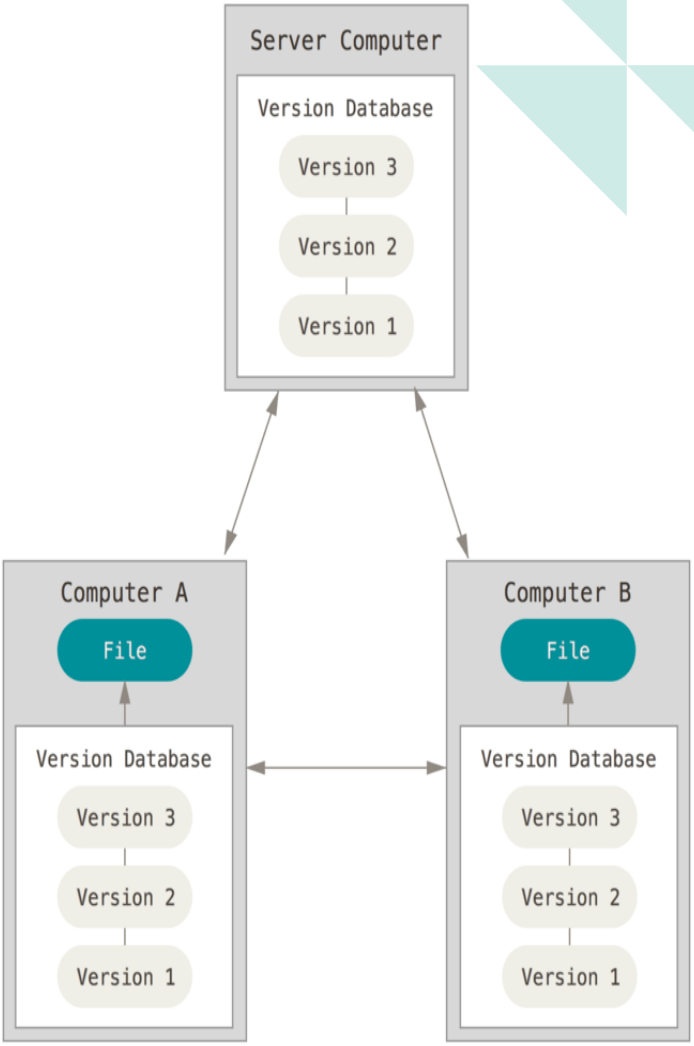


Esta configuración ofrece muchas ventajas, especialmente frente a VCS locales. Por ejemplo, todas las personas saben hasta cierto punto en qué están trabajando los otros colaboradores del proyecto. Los administradores tienen control detallado sobre qué puede hacer cada usuario, y es mucho más fácil administrar un CVCS que tener que lidiar con bases de datos locales en cada cliente.

Pero, esta configuración también tiene serias desventajas. La más obvia es el punto único de fallo que representa el servidor centralizado. Si ese servidor se cae durante una hora, entonces durante esa hora nadie podrá colaborar o guardar cambios en archivos en los que hayan estado trabajando. Si el disco duro en el que se encuentra la base de datos central se corrompe, y no se han realizado copias de seguridad adecuadamente, se perderá toda la información del proyecto, con excepción de las copias instantáneas que las personas tengan en sus máquinas locales.

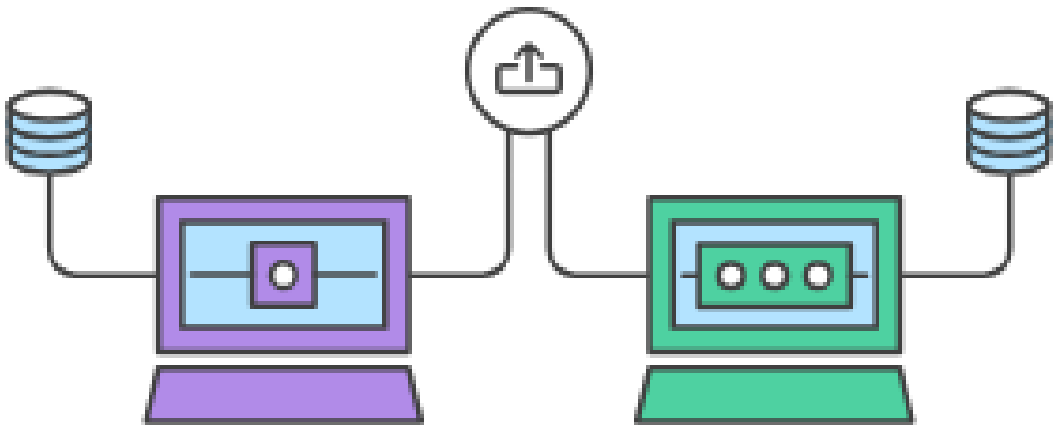
Sistemas de Control de Versiones Distribuidos: Los sistemas de Control de Versiones Distribuidos (DVCS por sus siglas en inglés) ofrecen soluciones para los problemas que han

sido mencionados. En un DVCS (como Git, Mercurial, Bazaar o Darcs), los clientes no solo descargan la última copia instantánea de los archivos, sino que se replica completamente el repositorio. De esta manera, si un servidor deja de funcionar y estos sistemas estaban colaborando a través de él, cualquiera de los repositorios disponibles en los clientes puede ser copiado al servidor con el fin de restaurarlo. Cada clon es realmente una copia completa de todos los datos.



Además, muchos de estos sistemas se encargan de manejar numerosos repositorios remotos con los cuales pueden trabajar, de tal forma que puedes colaborar simultáneamente con diferentes grupos de personas en distintas maneras dentro del mismo proyecto. Esto permite establecer varios flujos de trabajo que no son posibles en sistemas centralizados, como pueden ser los modelos jerárquicos.

"Commits" y "pull requests" en GitHub



Primero es importante saber que significan estas palabras "commits" y "pull requests" y por esta razón lo detallaremos a continuación:

Un **COMIT** es el primer lenguaje de programación para la manipulación de cadenas y de reconocimiento de patrones, el cual fue desarrollado por Yngve del MIT.

En Git, un commit es una especie de respaldo del estado actual de tu proyecto en el repositorio local, incluyendo todos los cambios que se le han hecho a cada segundo. En el contexto de la ciencia de la computación y la gestión de datos, hacer un commit se refiere a la idea de confirmar un conjunto de cambios provisionales de forma permanente.

Un **pull request** es una petición para integrar cambios de código a un proyecto, y se usa para contribuir a un proyecto grupal o de código abierto, o para proponer mejoras o correcciones. Un pull request puede incluir validaciones automáticas o manuales, asignación de tareas, despliegues, etc.

Es importante también conocer ¿Qué es hacer un forking? Cuando nos gusta el repositorio de alguien y nos gustaría tenerlo en nuestra cuenta de GitHub, hacemos un fork o bifurcación para poder trabajar con él en forma separada.

Cuando hacemos un fork de un repositorio, obtenemos una instancia de todo el repositorio con todo su historial. Luego, podemos hacer lo que queramos sin afectar la versión original.

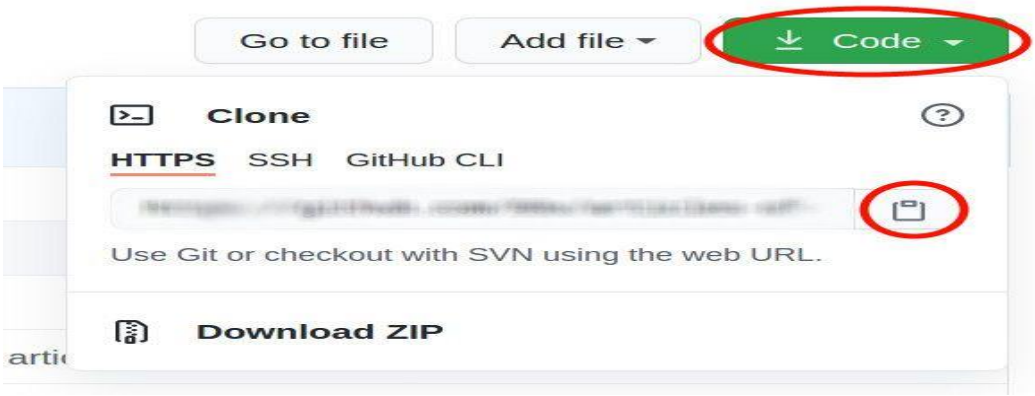
Ahora bien, realizaremos "commits" y "pull requests" en GitHub:

1. Realicemos un fork del repositorio

Realiza un fork del repositorio haciendo un clic en el botón fork de la parte superior de la página. Esto creará una instancia del repositorio completo en tu cuenta.

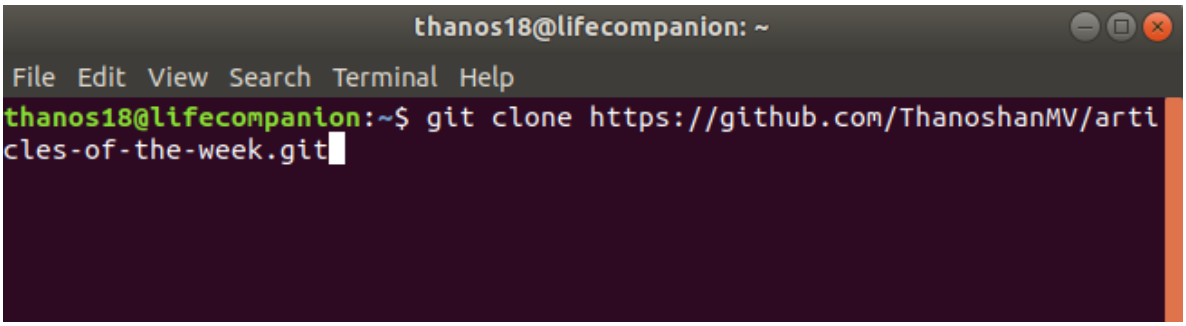
2. Clona el repositorio

Una vez que el repositorio esté en tu cuenta, clónalo a tu computador para trabajarlo localmente. Para clonarlo, has clic en el botón "Code" y copia el link.



Abre la terminal y ejecuta el siguiente comando. Esto clonará el repositorio localmente.

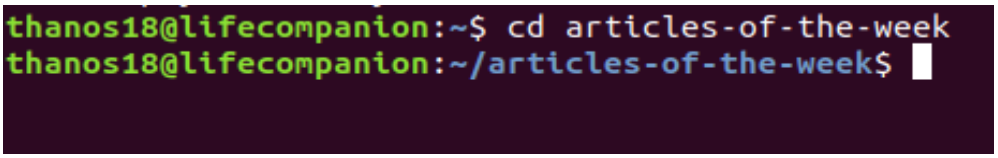
- $ git clone [DIRECCIÓN HTTPS]



Ahora hemos configurado una copia de la rama maestra desde el repositorio principal del proyecto en línea.

Debemos ir al repositorio clonado ejecutando el siguiente comando:

- $ cd [NOMBRE DEL REPOSITORIO]



3. Crea una rama

Es una buena práctica crear una rama (branch) nueva cuando trabajas con repositorios, ya sea que se trate de un proyecto pequeño o estés contribuyendo en un equipo de trabajo.El nombre de la rama debe ser breve y debe reflejar el trabajo que estamos haciendo.

Ahora crea una rama usando el comando git checkout:

- $ git checkout -b [Nombre de la Rama]

```
thanos18@lifecompanion:~/articles-of-the-week$ git checkout -b my-article
Switched to a new branch 'my-article'
thanos18@lifecompanion:~/articles-of-the-week$
```

4. Realiza cambios y confírmalos

Has cambios esenciales al proyecto y guárdalos.

Luego ejecuta git status, y verás los cambios.

```
thanos18@lifecompanion:~/articles-of-the-week$ git status
On branch my-article
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   2020-January/W2/README.md

no changes added to commit (use "git add" and/or "git commit -a")
thanos18@lifecompanion:~/articles-of-the-week$
```

Agrega esos cambios a la rama recién creada usando el comando git add:

- $ git add .

```
thanos18@lifecompanion:~/articles-of-the-week$ git status
On branch my-article
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   2020-January/W2/README.md

no changes added to commit (use "git add" and/or "git commit -a")
thanos18@lifecompanion:~/articles-of-the-week$ git add .
thanos18@lifecompanion:~/articles-of-the-week$
```

Ahora confirma esos cambios utilizando el comando git commit:

- $ git commit -m "Adding an article to week 02 of articles of the week"

```
thanos18@lifecompanion:~/articles-of-the-week$ git commit -m "Adding an articl
e to week 02 of articles of the week"
[my-article ae948c5] Adding an article to week 02 of articles of the week
 1 file changed, 1 insertion(+), 1 deletion(-)
thanos18@lifecompanion:~/articles-of-the-week$
```

5. Envía los cambios a GitHub

Para enviar los cambios a GitHub, debemos identificar el nombre del repositorio remoto.

- $ git remote

```
thanos18@lifecompanion:~/articles-of-the-week$ git remote
origin
thanos18@lifecompanion:~/articles-of-the-week$
```

Para este repositorio el nombre es "origin".

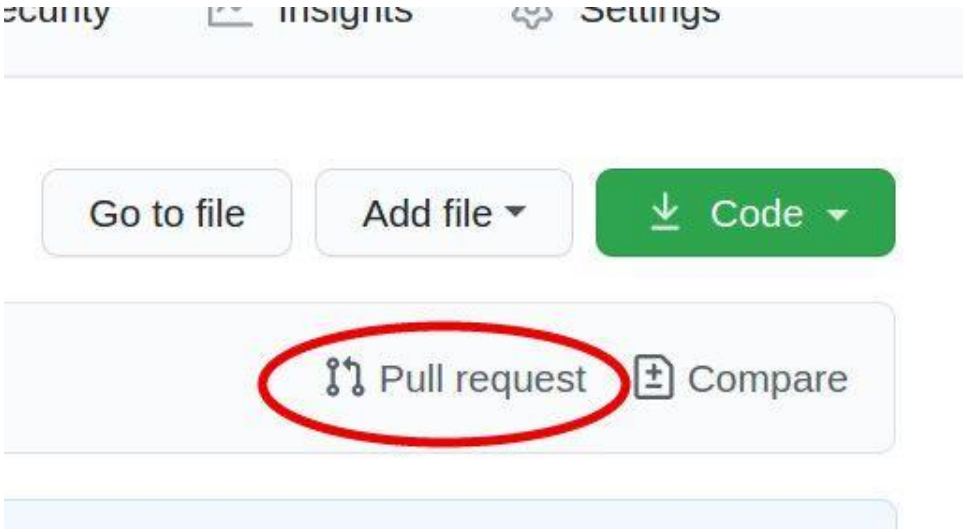
Luego de identificar el nombre podemos enviar en forma segura los cambios a GitHub.

- git push origin [Nombre de la Rama]

```
thanos18@lifecompanion:~/articles-of-the-week$ git push origin my-article
Username for 'https://github.com': ThanoshanMV
Password for 'https://ThanoshanMV@github.com':
Counting objects: 5, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 542 bytes | 542.00 KiB/s, done.
Total 5 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
remote:
remote: Create a pull request for 'my-article' on GitHub by visiting:
remote:      https://github.com/ThanoshanMV/articles-of-the-week/pull/new/my-a
rticle
remote:
To https://github.com/ThanoshanMV/articles-of-the-week.git
 * [new branch]      my-article -> my-article
thanos18@lifecompanion:~/articles-of-the-week$
```

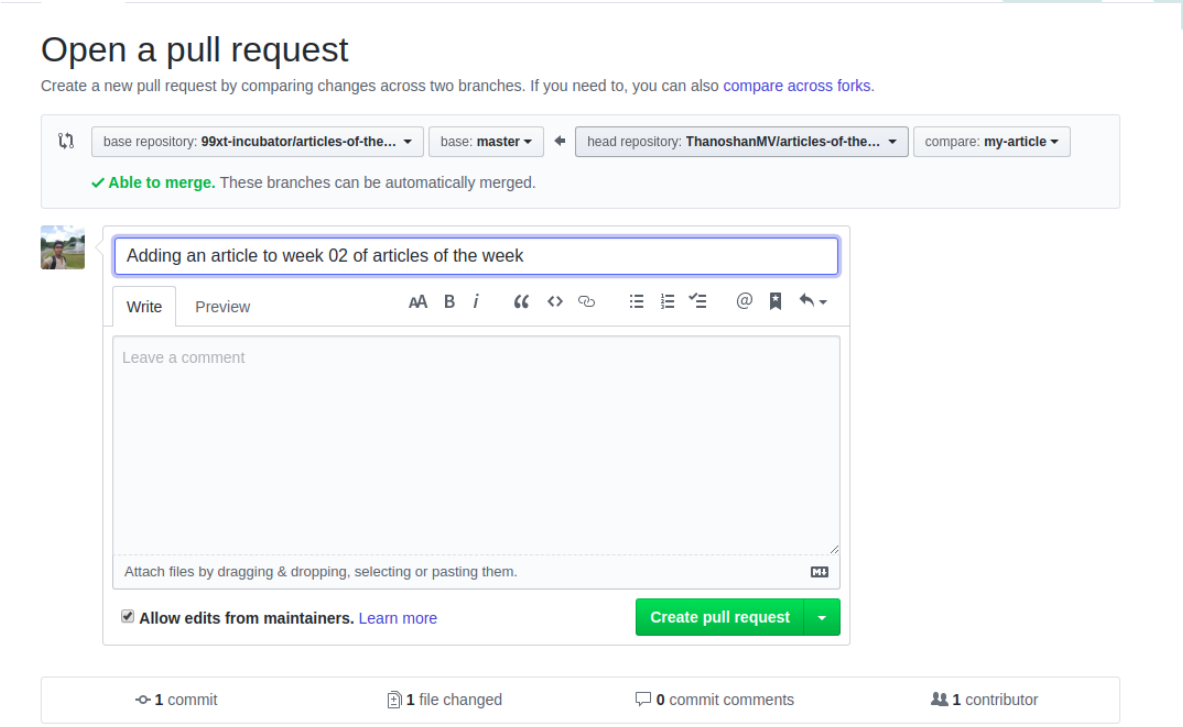
6. Crea un pull request

Ve a tu repositorio en GitHub y verás un botón llamado "Pull request", has clic en él.



Por favor, provee todos los detalles necesarios de lo que has hecho (puedes referenciar problemas utilizando "#"). Ahora, envía el pull request.

¡Felicitaciones! Tenemos nuestro primer pull request.



Si tu pull request es aceptado recibirás un mail.

7. Sincroniza tu rama maestra con la del repositorio original

Antes de enviar cualquier pull request al repositorio original debes sincronizar tu repositorio con aquel.

Incluso si no vas a enviar un pull request al repositorio original, es mejor efectuar la sincronización, ya que pueden haberse agregado algunas prestaciones o funciones adicionales y haberse corregido algunos errores desde la vez que realizaste un fork de aquel repositorio.

Sigue estos pasos para actualizar/sincronizar aquellos cambios con tu rama maestra:

A. Primero, revisa en que rama estás ubicado.

$ git branch

```
thanos18@lifecompanion:~/articles-of-the-week$ git branch
  master
* my-article
thanos18@lifecompanion:~/articles-of-the-week$
```

Esto enumerará todas las ramas e indicará en verde la rama actual o activa.

B. Cambia a la rama maestra.

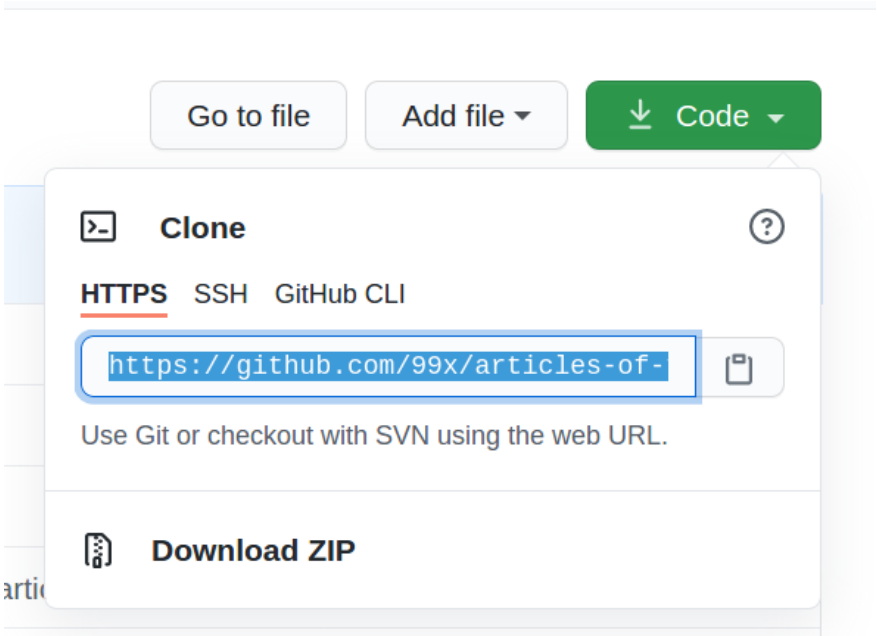
$ git checkout master

```
thanos18@lifecompanion:~/articles-of-the-week$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
thanos18@lifecompanion:~/articles-of-the-week$
```

C. Agrega el repositorio original como un repositorio upstream.

Para poder extraer los cambios desde el repositorio original a tu versión local, necesitas agregar el repositorio Git original como un repositorio upstream.

$ git remote add upstream [HTTPS]
Aquí, [HTTPS] es el URL que debes copiar del repositorio del propietario.




```
thanos18@lifecompanion:~/articles-of-the-week$ git remote add upstream https://github.com/
99xt-incubator/articles-of-the-week.git
```

D. Busca (fetch) el repositorio.

Busca todos los cambios del repositorio original. Las confirmaciones (commits) del repositorio original serán almacenadas en una rama local llamada upstream/master.

$ git fetch upstream

```
thanos18@lifecompanion:~/articles-of-the-week$ git fetch upstream
From https://github.com/ThanoshanMV/articles-of-the-week
 * [new branch]      master     -> upstream/master
 * [new branch]      my-article -> upstream/my-article
thanos18@lifecompanion:~/articles-of-the-week$
```

E. Fusiónala.

Fusiona los cambios de la rama upstream/master a tu rama maestra local. Esto hará que tu rama maestra se sincronice con el repositorio upstream sin perder tus cambios locales.

$ git merge upstream/master

F. Envía (push) los cambios a GitHub

En este punto tu rama local está sincronizada con la rama maestra del repositorio original. Si deseas actualizar el repositorio de GitHub, necesitas enviar tus cambios.

$ git push origin master

NOTA: Luego de sincronizar tu rama maestra puedes eliminar el repositorio upstream, si lo desea. Pero lo necesitará para actualizar/sincronizar tu repositorio en el futuro, por lo que es una buena práctica conservarlo.

```
thanos18@lifecompanion:~/articles-of-the-week$ git remote
origin
upstream
thanos18@lifecompanion:~/articles-of-the-week$ git remote rm upstream
thanos18@lifecompanion:~/articles-of-the-week$ git remote
origin
```

$ git remote rm [Nombre del Repositorio Remoto]

8. Elimina ramas innecesarias

Las ramas son creadas para propósitos especiales. Una vez que ese propósito se cumple, aquellas ramas ya no son necesarias, por lo que puedes eliminarlas.

$ git branch -d [Nombre de la Rama]

```
thanos18@lifecompanion:~/articles-of-the-week$ git branch
* master
  my-article
thanos18@lifecompanion:~/articles-of-the-week$ git branch -d my-article
Deleted branch my-article (was ae948c5).
thanos18@lifecompanion:~/articles-of-the-week$ git branch
* master
thanos18@lifecompanion:~/articles-of-the-week$
```

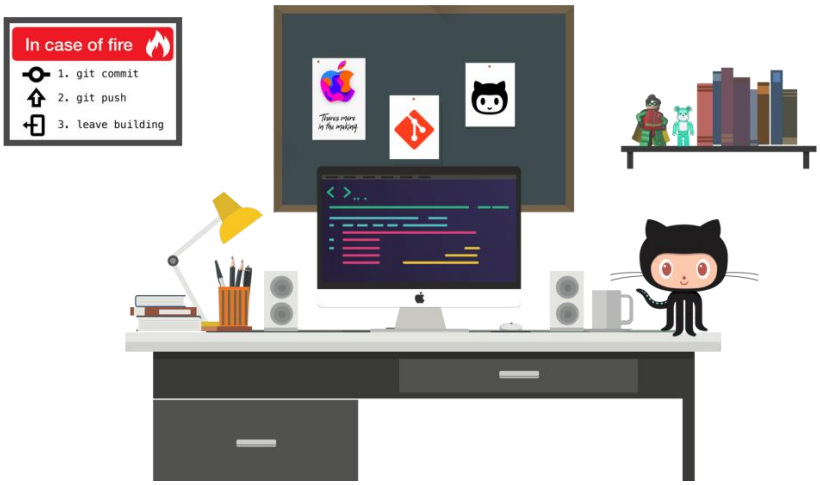
También, puedes eliminar su versión en GitHub.

git push origin --delete [Nombre de la Rama]

```
thanos18@lifecompanion:~/articles-of-the-week$ git push origin --delete my-article
Username for 'https://github.com': ThanoshanMV
Password for 'https://ThanoshanMV@github.com':
To https://github.com/ThanoshanMV/articles-of-the-week.git
 - [deleted]         my-article
thanos18@lifecompanion:~/articles-of-the-week$
```

Y de esta forma Ilustramos cómo se pueden realizar "commits" y "pull requests" en GitHub.

Integración de Mockups y GitHub en el Desarrollo Web



La palabra wireframe o muckups en el diseño web es un dibujo a mano alzada que permite definir, de forma gráfica, todas las pantallas que corresponden al funcionamiento de la web. Es una forma ágil para poder coordinar con el cliente, de forma visual, todo lo que nuestro visitante puede encontrarse durante el recorrido en nuestro desarrollo web.

No es tan importante definir en este paso el diseño real que va a tener la web, sino lo que es importante es que definamos todas las acciones o funciones que se va a encontrar el usuario cuando aterrice en esa página. Por ejemplo, si estamos definiendo un listado de productos en un ecommerce o tienda virtual, es importante definir para cada elemento del listado que vamos a mostrar al usuario: Título del producto, fotografía, precio, pequeña descripción, botón de favoritos, botón de añadir al carrito, valoración, etc. Todos los elementos que definan cada caja de producto, aunque no sea su ubicación definitiva. Si tendremos un buscador, que campos de filtros aparecerán, si tendremos paginación, etc. O si estamos en la página de producto, que elementos van a mostrarse en este detalle.

Una vez acordado y revisado con el cliente estas interfaces, es cuando pasaremos a la fase de diseño.

Conclusiones

En definitiva, GitHub es una poderosa herramienta para controlar el historial de versiones.

Todos pueden contribuir a proyectos de código abierto mediante pull requests. Las contribuciones no siempre son un código; también hay otras formas de contribuir. Y esto permite que nuestro proyecto al realizado de manera conjunta o grupal sea mucho más fácil la realización de este, permitiéndonos presentar un proyecto de manera mas ordenada y con una mejor calidad; y esto como consecuencia nos permitirá presentar paginas web con una mejor calidad y atracción para nuestros clientes.

En definitiva, usar todas estas herramientas nos facilitan la creación de proyectos web y la armoniosa colaboración de todos sus participantes, guiándose a resultados placenteros para los desarrolladores y los clientes.

Bibliografía

https://comparaiso.es/manuales/que-es-rcs

https://birdcom.es/blog/que-es-un-mockup/

https://techonlivesv.com/github-estrena-aplicacion-movil-que-ofrece-todo-el-soporte-para-la-creacion-de-software/

https://www.webempresa.com/hosting/que-es-github.html

https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Acerca-del-Control-de-Versiones

https://www.creativosonline.org/10-herramientas-imprescindibles-para-crear-mockups.html

https://tekla.io/blog/que-es-un-mockup-y-para-que-sirve/

https://www.atlassian.com/es/git/tutorials/making-a-pull-request

https://www.nocountryforgeeks.com/pull-requests/

https://www.freecodecamp.org/espanol/news/como-hacer-tu-primer-pull-request-en-github/

https://owius.com/mockups-en-el-diseno-web/