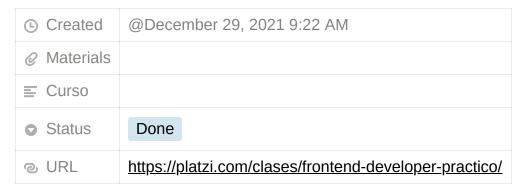
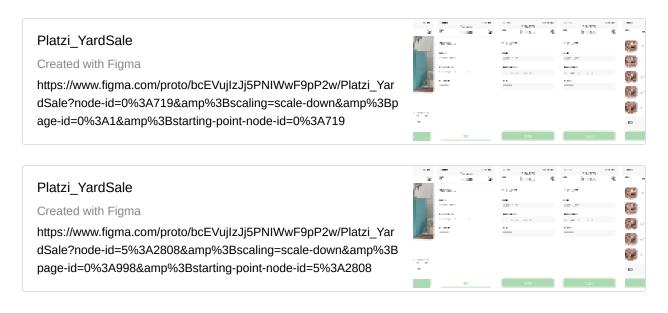
Curso Práctico de Frontend Developer



Bases diseño propuesto para el proyecto:



Prototipos Mobile y Desktop:



Sistema de diseño, assets y variables CSS

- Las variables permiten agilizar cambios en colores, tamaño de fuentes, etc. Van dentro del HEAD / style en una pseudoclase (:) que llamo "root".
 - o para definirlas, antecedo su nombre de "--"
- Fuentes las puedo descargar de "GoogleFonts". La idea es tener grosores diferentes: una muy delgada (ej: 300); otra media (ej: 500); y otra gruesa (ej: 700)
 - Descargo link's y los pego en el HEAD antes de "style" y después de los 3 primeros "meta".
 - Debo también indicar la familia dentro de CSS (en el "style"): con un tipo "body"
 pegando lo q me da GoogleFonts en "CSS rules"
- Link útil para orientarse / inspirarse en cuanto a los sistemas de diseño (tamaños, cantidad y tipo de colores, separaciones, etc)

Design - Shopify Polaris

A clean, simple style makes things feel approachable and efficient. Elegant code and lightweight assets means pages load more quickly. Built for flexibility, design tokens and new infrastructure let



https://polaris.shopify.com/design/design

- Los ASSETS son los logos e íconos q ubico en el directorio donde está el código
- Previo a escribir el código de HTML y CSS debe haber una maqueta (wireframes)
 hecha a través de otro medio, la cual guía lo q se hará en el código (qué cosas se
 deben crear y cómo se deben visualizar)
 - Ahí es fundamental definir cómo se pueden agrupar las cosas y en fxn de esto, determinar las propiedades de cada parte
 - Se inicia creando las etiquetas de HTML y luego se estilizan con CSS
 - Es muy útil ver en simultánea qué efecto tiene activar o anular determinadas cosas en el HTML y CSS

- Conforme el código se hace más extenso, se vuelve más complejo conectar lo q está en el HTML con el CSS. En ese caso es útil seleccionar xej la clase q se necesita en el HTML + Cmd + F y eso direcciona al style específico q controla esa parte.
- Se comienza a desarrollar pantalla por pantalla, en función de diferentes módulos q compondrán la página (ej: autenticación)
 - Cada pantalla puede tener sub-pantallas q se activan según las interacciones del usuario.
- Cuando reutilizo código de otras pantallas:
 - Eliminar lo q no se necesita
 - Si se requiere cambiar el nombre de algo, con sucesivos "Cmd + D" se selecciona automática/ dentro de cada bloque donde se repite ese nombre
- Cuando quiero replicar cosas en varias tarjetas (ej: poner la misma imagen en una etiqueta)
 - 1. Cmd + F
 - 2. Escribo la parte q quiero reemplazar
 - 3. click a flecha izquierda para desplegar
 - 4. reescribo lo q necesito
 - 5. click a opción en la derecha para reemplazar todo
- Si guiero "des indentar" bloques de código uso "Shift + Tab"

Buena práctica*

Si se quiere trabajar más adelante con JS, está bien escribir dentro del HTML los estilos del CSS usando la etiqueta style (ej: facilita separar y unir el código cuando se usa React.js).

En el caso de este curso, al ser el inicio de diferentes rutas, se optó por juntarlos en el mismo archivo. Pero si la versión final (la q se publicará y será usada por usuarios

reales) solo necesita de HTML y CSS, entonces sí es una buena práctica separar los estilos en archivos .CSS.

*: conjunto de costumbres, acciones, decisiones y/o herramientas que agilizan, mejoran el rendimiento, legibilidad, mantenimiento y/o escalabilidad de un proyecto en contexto específico.

Link's útiles para trabajar con "flex" y "grid"

- https://css-tricks.com/snippets/css/a-guide-to-flexbox/
- https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox
- https://css-tricks.com/snippets/css/complete-guide-grid/
- https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout