

DOCUMENTACIÓN JS SHOPPING CART

18 - 25 NOVIEMBRE 2019 / ROSER REVERTE

INTRODUCCIÓN

DESCRIPCIÓN DEL PROYECTO

En este proyecto la compañía necesita desarrollar un carrito de la compra usando exclusivamente Javascript. Además, es necesario que se guarde el contenido del carrito en el navegador del cliente, por lo que es indispensable usar localStorage para llevar a cabo la implementación.

TAREAS A REALIZAR

TAREA	PRIORIDAD	DIFICULTAD	TIEMPO ESTIMADO
Documentación de requisitos	ALTA	MEDIA	190min (Inicio, durante y final)
Analizar el documento y sus requisitos	ALTA	MEDIA	20min
Definición de tareas	ALTA	ALTA	30min
Crear repositorio en GitHub	ALTA	BAJA	10min
Búsqueda sobre localStorage	ALTA	MEDIA	30min
Búsqueda sobre OOP y analizar la organización de directorios y ficheros	ALTA	ALTA	90min
Investigar cambio de moneda entre € y \$	ALTA	ALTA	60min
Crear archivos HTML	ALTA	BAJA	20min

Crear header	MEDIA	MEDIA	20min
Crear footer	MEDIA	MEDIA	20min
Crear ficheros de clases	ALTA	ALTA	90min
Crear fichero de producto	ALTA	ALTA	30min
Lógica para cargar dinámicamente los productos en JS a HTML	ALTA	ALTA	40min
Guardar el producto en localStorage al seleccionarlo	ALTA	MEDIA	40min
Actualizar el precio según tamaño seleccionado	ALTA	ALTA	60min
Añadir “bagde” del cart	MEDIA	ALTA	40min
Stock: calcular stock disponible según los productos seleccionados	ALTA	ALTA	120min
Mostrar los productos del carrito	ALTA	ALTA	60min
Borrar productos del carrito	ALTA	ALTA	60min
Cambiar la moneda	MEDIA	ALTA	120min
Test	MEDIA	BAJA	20min
Revisar lo hecho el dia anterior y solucionar posibles errores.	MEDIA	???	60min
Realizar README	MEDIA	BAJA	10min
Revisar documentación	BAJA	MEDIA	30min
Imprevistos	ALTA	???	UN DIA

REGISTRO DE INCIDENCIAS DURANTE EL PROYECTO

INCIDENCIA	RESOLUCIÓN
No hacía falta el uso de compiladores	Redistribuir las tareas y volver a estructurar la organización y la estructura del proyecto.
Correcto uso de OOP	Estudiar, analizar y refactorizar la organización del código, ficheros y carpetas.
Actualización dinámica de las propiedades del producto	Re-inicializando las opciones dentro de el selector de tamaño y cantidad.
Guardar el producto en localStorage cuando lo añades al carrito	Añadiendo dentro del modelo de datos del producto, la propiedad: "selectedProperties".
Actualización del precio según moneda	Crear un "custom event".

CALENDARIO DEL PROYECTO

FECHA	FRANJA MAÑANA 8:00-14:00H	FRANJA TARDE 15:00-17:00H (++)
L 18/11	<p>Analizar el proyecto (y pills) y sus requisitos</p> <p>Definición de tareas</p> <p>Analizar y entender que es lo que aporta más valor al usuario</p> <p>Crear repositorio en GitHub</p>	<p>Búsqueda previa y aprendizaje sobre P00</p> <p>Analizar de qué forma organizaras el proyecto a nivel de directorios y ficheros</p> <p>Búsqueda sobre compiladores y setup. Hacer un "hello world".</p>
M 19/11	<p>Crear nuevo repositorio en GitHub</p> <p>Estudio de OOP en JS para ES5</p>	<ul style="list-style-type: none"> Definir estructura html Escoger assets, imagen general. Plantear y organizar la estructura de la lógica para el proyecto: crear clases y archivo de los productos

	Definir estructura html	
MC 20/11	<ul style="list-style-type: none"> • Crear documento js principal para mostrar contenido dinámicamente en index.html • Research contenido dinámico (precio, stock) 	<ul style="list-style-type: none"> • Guardar el id del producto en localStorage • Revisar documentación OOP
J 21/11		
V 22/11		<ul style="list-style-type: none"> • Re-organizar carpetas y código según OOP
S 23/11		<ul style="list-style-type: none"> • Actualizar el precio según tamaño seleccionado • Añadir selector de tamaño
D 24/11	<ul style="list-style-type: none"> • Añadir botón de añadir al carrito. • Añadir más productos • Añadir badge • Stock: calcular stock disponible según los productos seleccionados • Guardar el producto en localStorage al seleccionarlo 	<ul style="list-style-type: none"> • Calcular dinamicamente el stock • Mostrar productos en cart • Borrar productos del carrito • Realizar cambio de moneda • Escribir README • Terminar documentación
L 25/11	<ul style="list-style-type: none"> • Revisar documentación y proyecto. <p>FECHA LÍMITE DE ENTREGA (9:00h)</p>	

LECCIONES APRENDIDAS

- Es importante tener claro qué se pide en el proyecto y cómo. Definir el mínimo entregable y construir sobre el.
- No dar por sentado nada, comprobar que lo aprendido está asimilado y que lo aplicado funciona correctamente a medida que se va avanzando.
- Los estilos del proyecto no eran prioritarios, así que lo he dejado para el final, priorizando el funcionamiento de este.
- Estar en contacto con los demás compañeros enriquece mucho. Tanto a nivel del proyecto como a nivel personal. En momentos de estancamiento, ver otras maneras de enfocar el “problema” ayuda mucho.
- Hay que dejar mucho margen a las incidencias, porque siempre hay.

- La P00 es abstracta y hay que dedicar suficiente tiempo antes de empezar con la realización del proyecto para entender como es (incluyendo las diferencias con la programación funcional y como se aplica).
- El uso de las “high order functions” y los “callbacks”.
- La documentación ayuda mucho en la organización durante el proyecto, para saber en qué punto estas, y hacia donde vas.

MÉTRICAS DE CALIDAD

- Hacer uso de debugger/console
- Estudiar los conceptos antes de implementarlos
- Utilizar comentarios en el código para un mejor registro
- Hacer commits claros y concisos para llevar una buena actualización y seguimiento del proyecto
- Comprobar lo hecho el día anterior.
- Comprobar que se están cumpliendo los criterios y los requisitos
- Hacer uso de las “good practices” en el código.
- Comprobar que todos los elementos que se van añadiendo funcionan correctamente.
- Comprobar que se ve correctamente y funciona en todos los navegadores (Chrome, Safari, Firefox i IE 11) durante la elaboración del proyecto.
- Validar el código durante la elaboración del proyecto.
- Considerar +20% del tiempo disponible para el proyecto para imprevistos

DOCUMENTACIÓN DE REQUISITOS

- Debes de usar GIT. Es importante que las indicaciones y los commits sean los suficientemente explícitos y concretos como para poder entender los cambios sin la necesidad de requerir de información adicional en la medida de lo posible.
- No puedes hacer uso de Frameworks o librerías de terceros para la parte de Javascript
- Documentar todos tus algoritmos
- Usar programación orientada a objetos

- Trabajar de forma adecuada con los precios en Euros € y Dólares \$ (puedes colocar el valor de conversión entre monedas de forma manual y simular un servicio de un tercero)ç
- Los productos disponibles aparecerán en un listado con su correspondiente acción para “Añadir al Carrito”.
- El usuario podrá realizar como mínimo las siguientes acciones:
 - Añadir productos al carrito
 - Eliminar productos del carrito
 - Modificar cantidad de unidades
 - No podrá añadir más cantidades de las que se encuentren disponibles (simular stock)
 - Podrá cambiar opciones del producto que pueden influir en el precio e imagen que se me muestra al usuario

DOCUMENTACIÓN DE RIESGOS

RIESGO	SOLUCIÓN
Pérdida del proyecto	Proyecto en GitHub actualizado
Pérdida del ordenador	Uso de otro ordenador y proyecto en GitHub actualizado
Imprevisto familiar	Re-organizar las tareas: su horario y su prioridad.
Posibilidad de no entregar el proyecto completo	Justificar y argumentar en la memoria de la documentación del proyecto los motivos y posibles mejoras para el futuro.
Posibilidad de pérdida del documento	Utilizar Google-Docs con auto-guardado.
Posibilidad de fallos en el momento	Aplicar las métricas de calidad durante la realización del proyecto. Establecer un tiempo límite de 24h

de la entrega	antes de la entrega, para la elaboración completa del proyecto.
Dispersión, estancamiento, baja productividad	Replantear tareas. Pedir ayuda. Seguir estrictamente la organización de tareas y el calendario del proyecto. Posibilidad de no entregar el proyecto finalizado.

DOCUMENTACIÓN DEL WORKFLOW UTILIZADO

En este caso, al realizar el proyecto de manera individual, mediante mi repositorio personal de GitHub, voy a utilizar **GitFlow Basic**: una sola rama (master) para ir subiendo los cambios definitivos que vaya realizando en el proyecto, usando commits concisos y sencillos.



DOCUMENTACIÓN DE HERRAMIENTAS UTILIZADAS

- Visual Studio Code - editor
- Slack - comunicación
- GitHub - repositorio
- Google Docs - documentación
- Google Fonts - tipografía
- Stackoverflow, Medium, W3School - dudas técnicas
- Youtube - tutoriales
- <https://make.wordpress.org/core/handbook/best-practices/inline-documentation-standards/javascript/>

- <https://dev.to/shijiezhou/top-10-javascript-patterns-every-developers-like-168p>
- https://www.w3schools.com/jsref/jsref_filter.asp
- https://www.w3schools.com/jsref/jsref_map.asp
- https://www.w3schools.com/jsref/jsref_reduce.asp
- https://www.w3schools.com/js/js_objects.asp
- <https://medium.com/javascript-training/beginner-s-guide-to-webpack-b1f1a3638460>
- <https://medium.com/@jeffrey.allen.lewis/the-ultimate-2018-webpack-4-and-babel-setup-guide-npm-yarn-dependencies-compared-entry-points-866b577da6a>

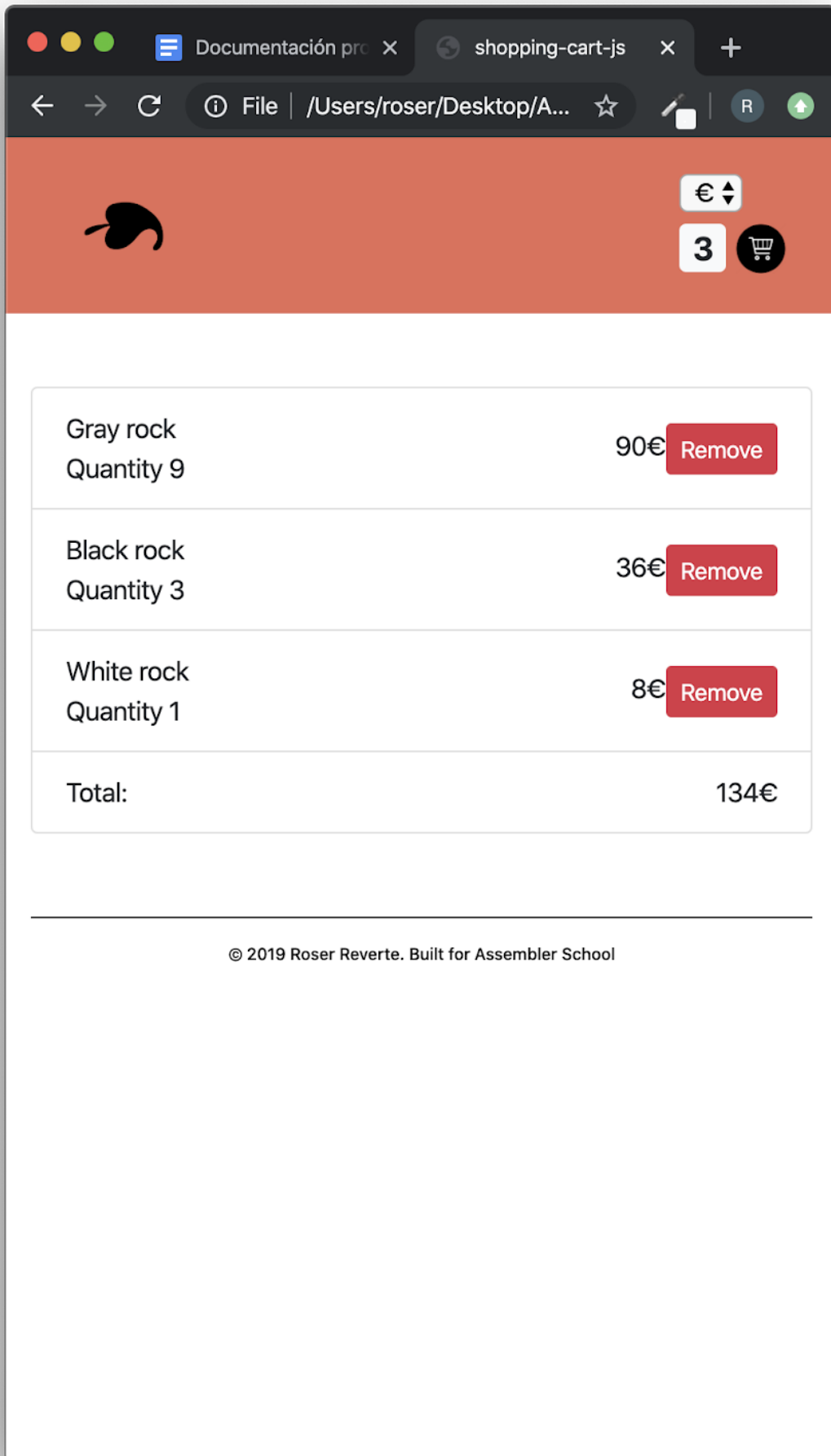
ANEXO-1

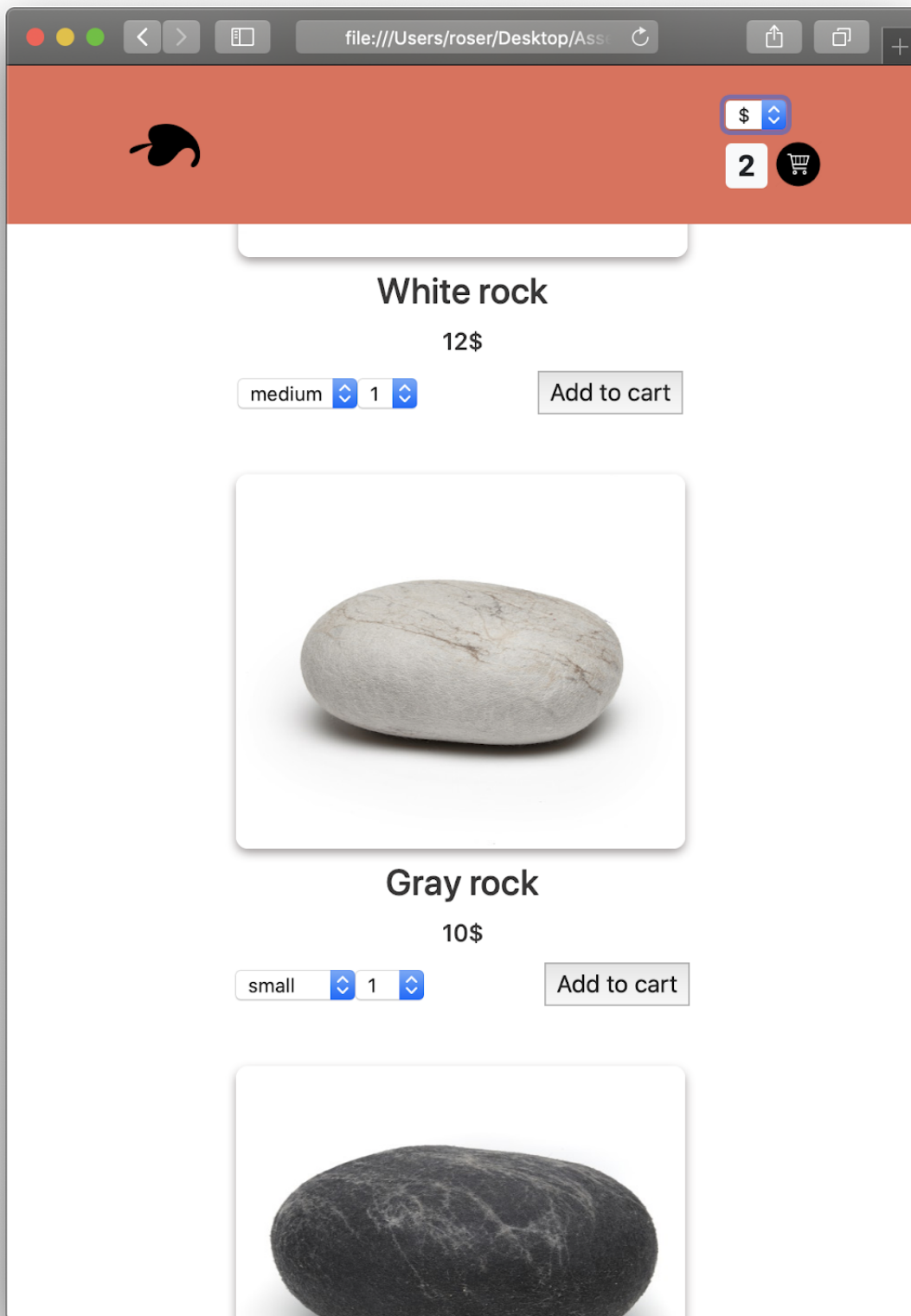
LINK REPOSITORIO DEL PROYECTO

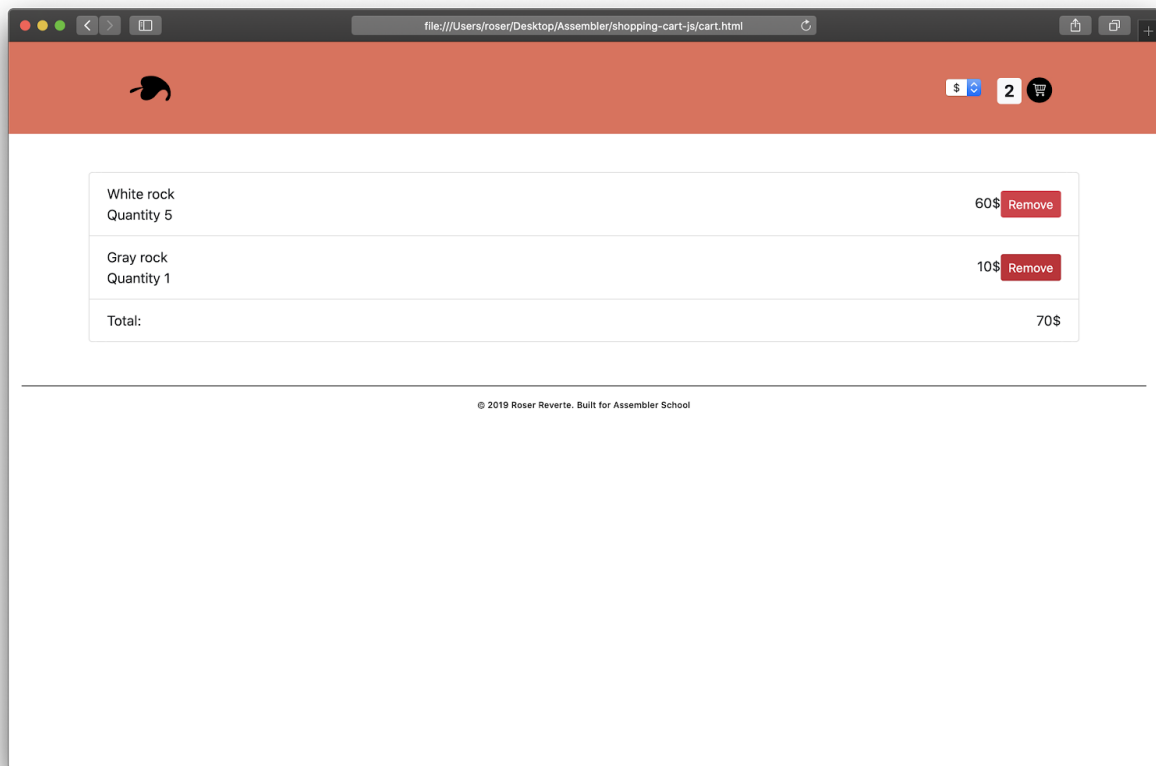
<https://github.com/reverroser/shopping-cart-js>

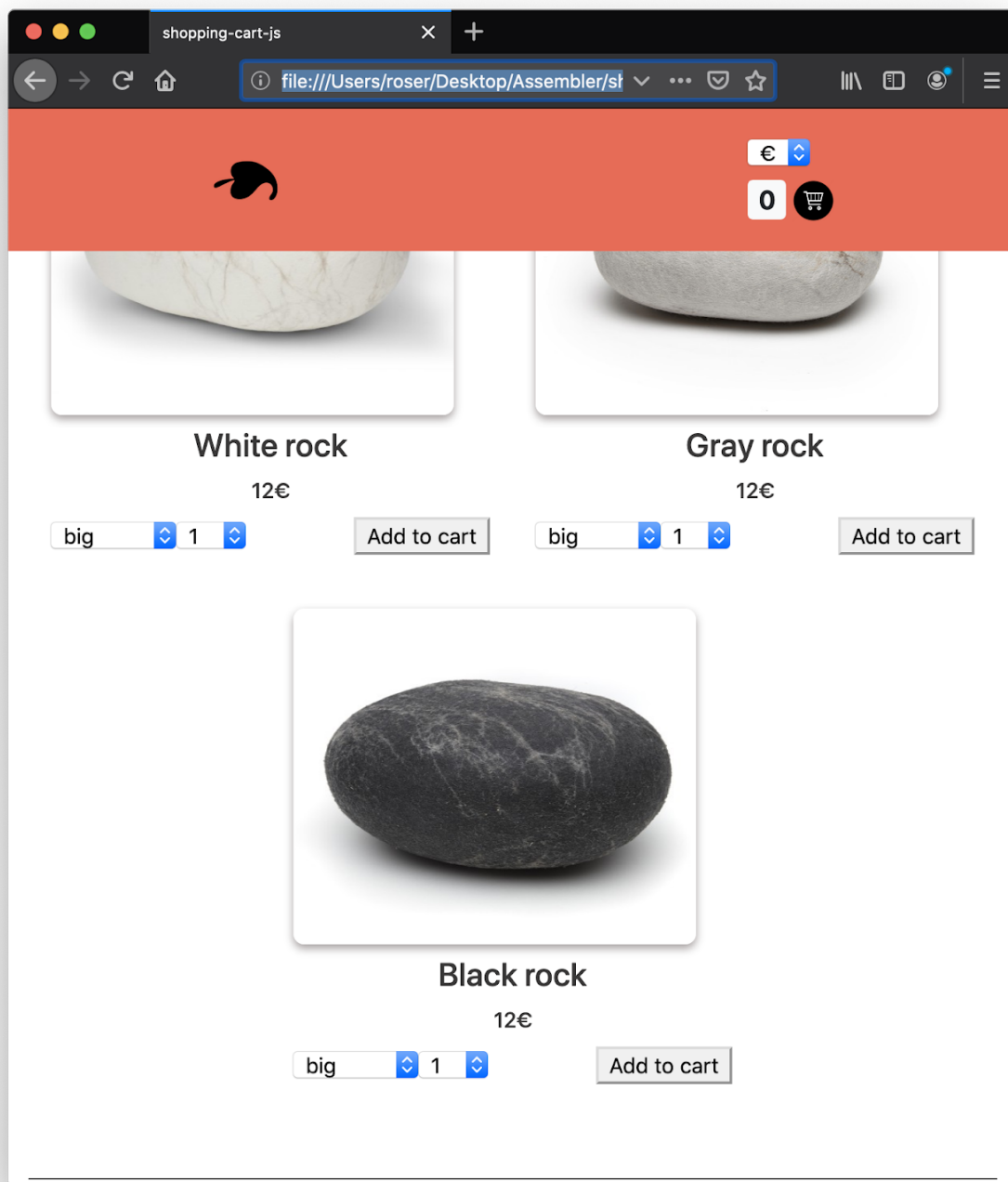
ANEXO-2

CAPTURAS DE PANTALLA NAVEGADORES











FacebookInstagram(31) Como UtilizarConfirmar correoreverroser/shoppDescargasshopping-cart-js


ArchivoC:/Users/kevin_000/Downloads/shopping-cart-js-master/index.htmlGoogleYouTubeConvertidor YouTubeNetflix - Ve series o...FacebookIniciar sesiónEdmodoInicioCampus Virtual UN...Página principal DIPP G Nula



0



SHOP




White rock

undefinednull

big

1

Add to cart




Gray rock

undefinednull

big

1

Add to cart



Black rock

undefinednull

big

1

Add to cart

© 2019 Roser Reverte. Built for Assembler School

