**Proceso instalación Visual Studio Code**

**Paso 1:** Escribimos en Google Visual Studio Code y seleccionamos donde dice “Download”.



**Paso 2:** Seleccionamos el sistema operativo que tenemos y lo descargamos.



**Paso 3:** Al darle clic nos descargará un .exe, al cual le daremos clic encima.



**Paso 4:** Lee y acepta el acuerdo de licencia. Haz clic en Next para continuar.



**Paso 5:** Puedes cambiar la ubicación de la carpeta de instalación o mantener la configuración predeterminada. Haz clic en Next para continuar.



**Paso 6:** Elige si deseas cambiar el nombre de la carpeta de accesos directos en el menú Inicio o si no deseas instalar accesos directos en absoluto. Haz clic en Next.



**Paso 7:** Selecciona las tareas adicionales, por ej. crear un icono en el escritorio o añadir opciones al menú contextual de Windows Explorer. Haz clic en Next.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

**Paso 8:** Haz clic en Install para iniciar la instalación.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Paso 9:** El programa está instalado y listo para usar. Haz clic en Finish para finalizar la instalación y lanzar el programa.



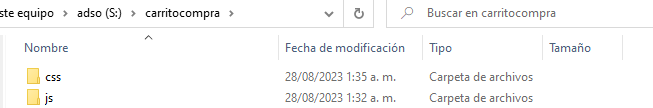
**Creación de archivos y carpetas**

**Paso 1:** Creamos una carpeta raíz. En nuestro caso la llamaremos “carritocompra”.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente con confianza media

**Paso 2:** Dentro de esta, crearemos otras dos carpetas llamadas “css”, la cual contendrá todo lo estético de nuestra página y otra llamada “js” la cual contendrá la lógica de nuestra página.



**Carrito de compras**

**Paso 1:** Para abrir el Visual Studio Code, haremos lo siguiente: Dentro de la carpeta raíz, en la barra superior, escribiremos cmd y le damos enter.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Paso 2:** Eso nos abrirá una terminal, solo tendremos que escribir “code .”.

Texto

Descripción generada automáticamente

**Paso 3:** Esto nos abrirá el visual studio code. Ahora, ya que estamos dentro, crearemos un archivo llamado “index.html” dándole click derecho a la barra de la izquierda y luego click izquierdo en “new file” o “nuevo archivo”

Captura de pantalla de un celular

Descripción generada automáticamente

**Paso 4:** Llamaremos el archivo “index.html”

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

**Paso 5:** Ahora crearemos nuestro archivo de estilos dándole click derecho en nuestra carpeta llamada “css” y click izquierdo en “new file”

Captura de pantalla de un celular

Descripción generada automáticamente

**Paso 6:** El archivo de llamara “style.css”

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Paso 7:** Ahora vamos a crear nuestro archivo js dándole click derecho en la carpeta “js” de la izquierda y luego click izquierdo en “new file”

Captura de pantalla de un celular

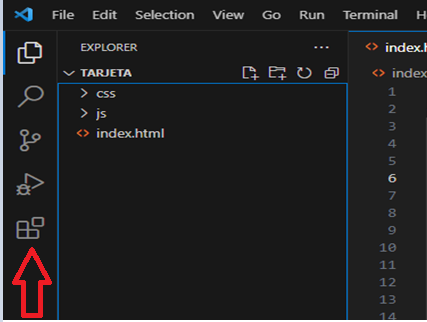
Descripción generada automáticamente

**Paso 8:** Llamamos el archivo como “app.js” y así tuvo que quedar nuestra estructuración de carpetas y archivos

Interfaz de usuario gráfica

Descripción generada automáticamente

**Paso 9:** Ahora para poder visualizar como se está viendo nuestro proyecto instalaremos una extensión de visual estudio code, dandole click al icono de abajo a la izquierda.

****

**Paso 10:** Nos saldrá esta pestaña y buscaremos live server, en la barra de búsqueda

**Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente**

**Paso 11:** Ahora que encontramos lo que necesitamos, le damos click a “install”

**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente**

**Paso 12:** Ahora que tenemos instalado el live server nos debería de aparecer algo así donde podremos volver a nuestro proyecto dando click al explorador.

**Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente**

**Paso 13:** Una vez te sale el explorador volvemos al Index.html

**Captura de pantalla de un celular

Descripción generada automáticamente**

**Paso 14:** Una vez estemos en el archivo index de nuevo vamos a colocar el código “!” y escogemos la primera opción.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Paso 15:** Ahora vamos a cambiar el titulo (línea 6) por “Carrito de compras”

Texto

Descripción generada automáticamente

**Paso 16:** Vinculamos nuestro css dentro del head (línea 8) para dentro de un futuro modificar los diseños

Texto

Descripción generada automáticamente

**Paso 17:** Dentro del body crearemos un div principal con el id “app” (línea 12) dentro de ese div creamos 2 divs más un con la clase “menu” (línea 14) y otro con el id “store-container”(línea 19) y por último creamos otro div dentro del div con la clase “menu” (línea 16) con el id “shopping-cart-container” y le ponemos la clase “hide”

Pantalla de computadora con letras

Descripción generada automáticamente

**Paso 18:** Ahora vinculamos nuestro javascript antes de terminar el body (línea 22)

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

**Paso 19:** Ahora iremos a nuestro archivo javascript “app.js” (Paso 8) y creamos un objeto para simular una base de datos llamado “db” (línea 2)

Texto

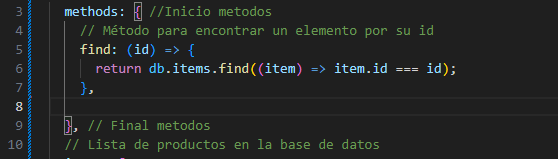
Descripción generada automáticamente

**Paso 20:** Ahora crearemos un contenedor para almacenar los metodos (línea 3)

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Paso 21:** Dentro de nuestro contenedor de métodos crearemos un método para encontrar un elemento por su id (línea 5)

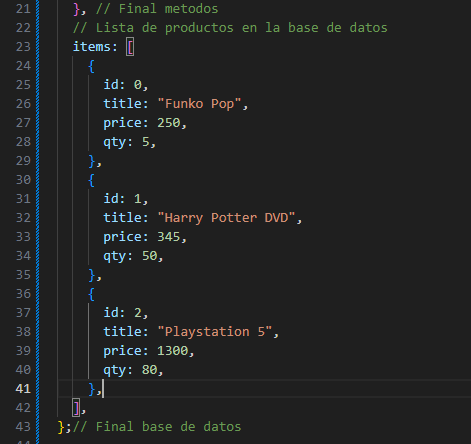


**Paso 22:** Ahora creamos un nuevo método para remover items (línea 9), recorremos todos los items (línea 11) encuentra el producto correspondiente (línea 13) y recude su cantidad (línea 15) y colocamos que se muestre actualizada en la consola (línea 19)

Captura de pantalla de computadora

Descripción generada automáticamente

**Paso 23:** Ahora creamos un array con los items que va a simular la base de datos. (línea 23) vamos a colocarles un id diferente a cada uno, también el título, el precio de ese objeto y la cantidad que hay.

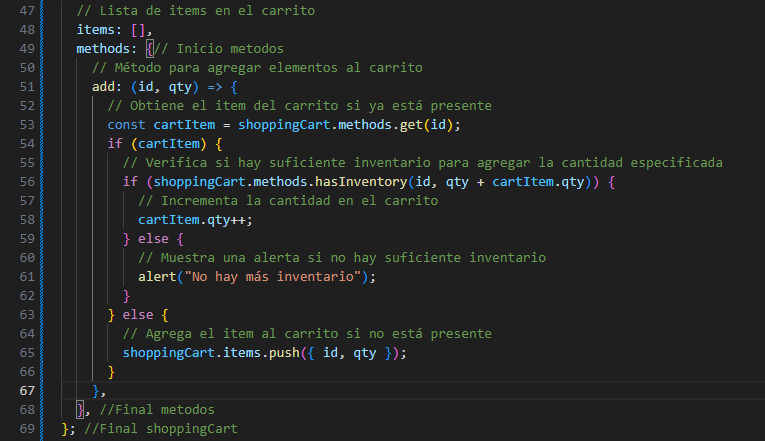


**Paso 24:** Ahora crearemos un objeto que va a representar el carrito de compras llamado “shoppingCart” (línea 46)

Texto

Descripción generada automáticamente

**Paso 25:** Dentro de “shoppingCart” vamos a crear un array con los items (línea 48) y comenzamos con los métodos (línea 49) primero vamos a añadir el método para agregar un item al carrito y lo llamaremos “add”, (línea 51) y adentro de este método vamos a mirar si ya esta presente el objeto en el carrito (línea 53) añadiendo un condicional(línea 54) y adentro otro condicional que mirara si hay suficientes items del objeto para añadirlo al carrito (línea 56) donde utilizaremos un método que aun no tenemos llamado “hasInventory” y en caso de que no haya en el inventorio que mande un mensaje que no hay objetos (línea 61) y en caso de que no este el item en el carrito vamos a añadirlo (línea 65)



**Paso 26:** Ahora vamos a crear un nuevo método el cual va a remover elementos del carrito (línea 69) lo llamaremos “remove” lo primero será obtener el id del carrito (línea 71) en caso de que la cantidad sea mayor a 1 lo resta (línea 74) y de lo contrario lo quita (línea 78)

Texto

Descripción generada automáticamente

**Paso 27:** Ahora un nuevo método para contar la cantidad de objetos que hay (línea 84), la llamaremos “count”

Texto

Descripción generada automáticamente

**Paso 28:** Ahora crearemos otro método llamado “get” (línea 89) para obtener un objeto del carrito (paso 89) donde se utilizara el código “.findIndex” (línea 91) para encontrar el objeto y en caso de que este en el carro lo retorna de lo contrario solo envía nulo.

Pantalla de computadora con letras

Descripción generada automáticamente con confianza media

**Paso 29:** Este método llamado “getTotal” (línea 96) funcionara para calcular el tota de la compra con una variable que se llamara “total” (línea 97) y haremos que total sea el precio multiplicado por la cantidad en el carro y retorne el total (línea 103)

Texto

Descripción generada automáticamente

**Paso 30:** Crearemos un nuevo método para saber si hay suficiente inventario para añadir al carro y lo llamaremos “hasInventory” (línea 106) y encontrara el objeto en la base de datos y mirara su cantidad, en caso de que sea mayor o igual a 0 al restar significa que hay inventario. (línea 108)

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

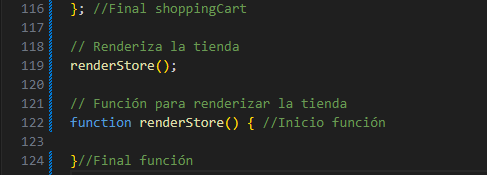
Descripción generada automáticamente

**Paso 31:** Y creamos el ultimo método llamado “purchase” y a su vez cerramos “shoppingCart” el cual habíamos comenzado en la línea 46 (paso 24)

Texto

Descripción generada automáticamente

**Paso 32:** Llamamos una función llamada “renderStore” (línea 119) y la creamos abajo (línea 122)



**Paso 33:** Ahora dentro de la función llamada “renderStore” que servirá para que se muestre nuestros objetos en la página vamos a generar elementos HTML por cada producto que hemos creado anteriormente con la función .map (línea 124) y luego vamos a generar el html (línea 126 a la 132)

Pantalla de computadora con letras

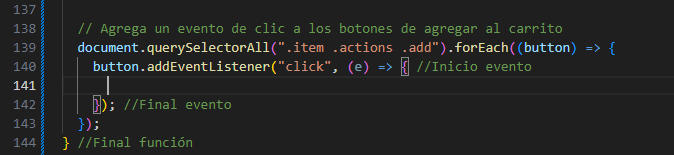
Descripción generada automáticamente con confianza media

**Paso 34:** Dentro de la función “renderStore” vamos a agregar los elementos al contenedor de la tienda (línea 136)

Interfaz de usuario gráfica, Texto, Sitio web

Descripción generada automáticamente

**Paso 35:** Dentro de la misma función “renderStore” vamos a agregar un evento click a los botones de agregar al carrito (línea 139)



**Paso 36:** Ahora dentro del evento vamos a obtener el id (línea 142) y encontrar el producto (línea 144) va a verificar si hay inventario y lo agrega al carrito (línea 147) y si lo hay actualiza el carrito con una función que crearemos ahora llamada “renderShoppingCart” (línea 150) en caso de que no haya va a enviar una alerta avisando que no hay existencias (línea 153)

Captura de pantalla de computadora

Descripción generada automáticamente

**Paso 37:** Ahora vamos a crear la función llamada “renderShoppingCart” que mencionamos anteriormente (línea 160)

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

**Paso 38:** En la función “renderShoppingCart” vamos a generar elementos html con los objetos que se añadan

Captura de pantalla de computadora

Descripción generada automáticamente

**Paso 39:** Ahora creamos unos elementos html para cerrar el carrito (línea 180) y otra para terminar la compra (línea 184) y una nueva línea que mostrara el total del precio (línea 191)

Texto

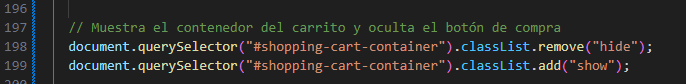
Descripción generada automáticamente

**Paso 40:** Ahora debajo de lo que muestra el total vamos a agregar lo elementos seleccionados al contenedor del carrito de compras (línea 194)

Texto

Descripción generada automáticamente

**Paso 41:** Ahora les añadimos una clase para mostrarlo y le quitamos la anterior (línea 198-199)

  
**Paso 42:** Ahora agregamos los eventos para añadir items al carrito y actualizamos el carrito llamando la función “renderShoppingCart” (línea 206)

Texto

Descripción generada automáticamente

**Paso 43:** Ahora agregamos el evento para remover los items (línea 210) y actualizamos el carrito de compras (línea 214)

Texto

Descripción generada automáticamente

**Paso 44:** Ahora vamos a añadir el evento para cerrar el carrito de compras

Texto

Descripción generada automáticamente

**Paso 45:** Y por último un evento para realizar la compra (línea 225) y que al apretar el botón active el evento de purchase “línea 228”

Texto

Descripción generada automáticamente

**Paso 46:** Y creamos una ultima función para convertir un número en formato de moneda.

Texto

Descripción generada automáticamente

**Paso 47:** Ahora vamos a mirar nuestro proyecto gracias a la extensión “live server” (paso 9 a 13) yendo a nuestro archivo “index.html” y daremos click derecho en cualquier parte del código luego click izquierdo en “open with live server”

Imagen que contiene Texto

Descripción generada automáticamente

**Paso 48:** Gracias a esto se abrirá una pestaña en Google y te mostrará como va nuestro proyecto.

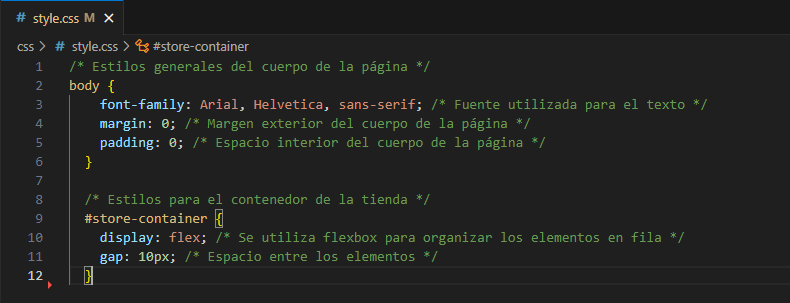
Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Paso 49:** Ahora para añadirle diseño a nuestra aplicación iremos a nuestro arhicov “style.css” y cambiaremos un poco el body y el contenedor de la tienda.



**Paso 50:** Ahora miramos como está quedando

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

**Paso 51:** Cambiamos el diseño de los items

Texto

Descripción generada automáticamente

**Paso 52:** Así han quedado nuestros objetos por ahora

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Paso 53:** Ahora cambiamos un poco el diseño de nuestro carrito de compras.

Texto

Descripción generada automáticamente

**Paso 54:** Así va de momento

Imagen que contiene Forma

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

**Paso 55:** Ahora cambiamos la clase para que sea visible o no.

Una captura de pantalla de un celular

Descripción generada automáticamente

**Paso 56:** Ahora a nuestros items les modificamos un poco el diseño poniendo el titulo con negrilla y demás cosas.

Pantalla de un video juego

Descripción generada automáticamente con confianza media

**Paso 57:** Así está quedando

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Paso 58:** Ahora cambiamos el diseño de los contenedores, el total y el botón.

Texto

Descripción generada automáticamente  
**Paso 59:** Así ha quedado finalmente nuestro proyecto.

Imagen que contiene Forma

Descripción generada automáticamente

Imagen que contiene Forma

Descripción generada automáticamente