RANDART

TPO Desarrollo Web HTML, CSS y JavaScript

Codo a Codo – Comisión 23529

Descripción: Página web que ofrece arte y coleccionismo al azar, dando a todos los usuarios la igual oportunidad de exhibir y vender sus obras en la plataforma.

Público objetivo: Artistas y coleccionistas que desean igualdad de oportunidades para tener visibilidad de sus trabajos y vender los mismos, así como compradores interesados en adquirir obras originales de artistas emergentes.

Repositorio: https://github.com/nrmattar/randart

Link de la web: https://randart.netlify.app/

Integrantes:

- Ruiz Mattar, Nahuel - DNI 33193110 (Representante)
- Esposito, Maximiliano Roberto - DNI 31224315
- López Becerra, Analía - DNI 35270596
- Ponce, Emanuel David - DNI 38644043

OBJETIVO DEL PROYECTO

El objetivo principal de Randart es proporcionar una plataforma en línea en la que los artistas y coleccionistas puedan exhibir sus obras y los usuarios puedan comprarlas o intercambiarlas de manera aleatoria. Además, se busca fomentar la creatividad y la cultura artística en línea.

PLANIFICACIÓN

El proyecto Randart ha sido desarrollado con la metodología ágil, lo que nos ha brindado una mayor flexibilidad y adaptabilidad a los cambios durante el proceso de desarrollo. Partimos de la idea de crear una plataforma web enfocada en el arte y a medida que avanzábamos en las

clases, fuimos proponiendo nuevos contenidos y aplicando lo aprendido para mejorar constantemente el proyecto.

Para asegurar una gestión efectiva del proyecto, hemos establecido un cronograma de trabajo y hemos asignado tareas específicas a cada miembro del equipo

ORGANIZACIÓN

El equipo de desarrollo de Randart está formado por cuatro miembros, cada uno con habilidades y conocimientos que complementan el proyecto. Para asegurar una comunicación constante y efectiva, hemos establecido un chat grupal donde discutimos las ideas y al menos dos reuniones semanales a través de Zoom. Esto nos permite colaborar de manera eficiente y garantizar una entrega exitosa del proyecto.

A medida que surgían nuevas ideas, las discutimos en conjunto y las incorporamos en un esquema detallado que incluía una lista de temas pendientes. Posteriormente, distribuiremos estos temas según las destrezas y conocimientos de cada miembro del equipo. Para facilitar el desarrollo individual, cada miembro tenía su propia rama (Branch) en GitHub, donde trabajaba en el tema asignado. Una vez finalizado, el representante del equipo se encargaba de unificar los cambios y asegurar la coherencia y calidad del proyecto.

Esta metodología de trabajo nos ha permitido mantener un flujo de trabajo eficiente, maximizando la productividad y asegurando la calidad del proyecto Randart.

ESTRUCTURA DEL SITIO

Páginas

La web tiene 4 páginas principales accesibles desde su menú de navegación y una página extra la cual está linkeada al detalle de cada obra desde la galería.

- Inicio (index.html): Introducción de la marca y visión aleatoria con reproducción automática de las obras principales
    - Diseño: randart.css y banner.css
    - Scripts: banner.js y menu.js

- Galería (galería.html): exposición de obras de arte (articulo.html). Al clickear cualquier obra se redirige hacia articulo.html enviando por parámetro la información de la obra seleccionada la cual es parseada vía javascript (getarticulo.js).

- Diseño: randart.css y articulos.css
- Scripts: script.js, getarticulo.js

- Articulo (articulo.html - No accesible via menu): Aqui se popula la información del artículo seleccionado en la pantalla anterior y se utiliza javascript con diversos propósitos: popular los precios de las obras en dólares y convertirlo a pesos vía API, parsear la información recibida por parámetros y construir dinámicamente el contenedor con la estructura de la información presentada.

- Directorio (directorio.html): guía de direcciones de exposiciones y galerías físicas utilizando iframes para la populacion de los mapas.
  - Diseño: randart.css y articulos.css
  - Scripts: menu.js
- Contacto (formulario.html): formulario de registro validado mediante JavaScript
  - Diseño: randart.css y formulario.css
  - Scripts: formulario.js

Diseño y estilo

El sitio tiene un estilo minimalista. Se utilizó una gama de colores que comprende blanco, #585858 (gris) y #f12554 (simil fucsia). Los elementos visuales que predominan son un menú, logotipo, footer con iconos de contacto y un cuerpo que varía según la página visitada. Desde un banner que ofrece una transición de las obras contenidas en la galería, así como una grilla con dichas obras, mapas de galerías de arte y un formulario de contacto.

Contenido y funcionalidades

Se desarrollaron en 4 archivos .JS

1. **banner.js** se emplearon varias funciones de validaciones y verificaciones diferentes para que el contendor reaccione, de manera que se pueda actualizar automáticamente al agregarse imágenes al archivo galería.html.
   a. El código comienza ejecutando el evento "DOMContentLoaded", lo que significa que se ejecutará una vez que el documento HTML esté completamente cargado en el navegador.
   b. Luego, se obtienen referencias a dos elementos HTML en el documento: ".gallery" y ".gallery-indicators", los cuáles son contenedores para las imágenes y los indicadores de la galería.

c. Se inicializan las variables, currentImageIndex (para rastrear la imagen actual) e imágenes (que se usará para almacenar las imágenes cargadas).
d. La función changeImage(index) se define para cambiar la imagen que se muestra en la galería. Oculta la imagen actual y muestra la imagen en el índice index, además de actualizar los indicadores.
e. La función updateIndicators() se utiliza para actualizar los indicadores de la galería, resaltando el indicador correspondiente a la imagen actual.
f. Luego, el código realiza una solicitud fetch("/templates/galeria.html"), para cargar el contenido de "galeria.html," que probablemente contiene las imágenes y otros elementos relacionados con la galería.
g. Cuando se completa la solicitud, el contenido se inserta en un elemento temporal (tempDiv) para facilitar la manipulación.
h. Se limpia el contenedor actual de imágenes (galleryContainer) para asegurarse de que esté vacío.
i. Se buscan todas las imágenes en el contenido de "galeria.html" y se almacenan en la variable images. Luego, se agregan estas imágenes al contenedor de la galería.
j. En el código también agrega un evento clic en el contenedor de imágenes para redirigir al usuario a la página de la galería "galeria.html."
k. Por último, se configura un temporizador para cambiar automáticamente la imagen cada 5 segundos. Esto se hace usando setInterval, que llama a la función changeImage para mostrar la siguiente imagen en la galeria.get artículos.js se emplearon varias funciones de validaciones y verificaciones diferentes para que el formulario reaccione según los datos ingresados por el usuario, y a su vez, guíe a este en una correcta forma de insertar la información requerida para proceder con él envió del mismo, con la información más veraz posible.

2. **Menu.js** se utiliza para que en los dispositivos móviles se visualice u oculten los botones para desplegar o colapsar el icono del menú. Dicho comportamiento se realiza agregando o removiendo subclases a los elementos (botones)

3. **Formulario.js** se emplearon varias funciones de validaciones y verificaciones diferentes para que el formulario reaccione según los datos ingresados por el usuario, y a su vez, guíe a este en una correcta forma de insertar la información requerida para proceder con el envió del mismo, con la información más veraz posible.
    a. En primera instancia se escribió más de una expresión de Javascript (document.getElement ById("nombre").value.trim();)) que se utiliza para acceder al elemento específico y que permita trabajar con la información registrada por el usuario, con el fin de que a posteriori se le apliquen las verificaciones pertinentes según los criterios y reglas ya pensados de antemano. document.getElementById("nombre"): Selecciona en este caso el elemento HTML cuyo atributo(id), en este caso, es “nombre”. Teniendo a document: como el objeto global del documento en sí mismo, getElementById: como el método para seleccionar el elemento buscado (de donde se toma el dato)

b. `.value`: Con esta propiedad se busca que lo ingresado o no por el usuario, sea captado para su posterior análisis contrastando con las reglas que se escribieron pensando en el tipo de formulario que se desea obtener.
c. `.trim()`: `.trim()` este método se utiliza para eliminar los espacios en blanco al principio y al final.

Validaciones para los distintos campos que se deben completar en el formulario

Validación de los campos/input en blanco:

Reglas utilizadas:

Nombre:

```
var nombreTest = /^([A-Za-zÑñÁáÉéÍíÓóÚú]+['\-]{0,1}[A-Za-zÑñÁáÉéÍíÓóÚú]+)(\s+([A-Za-zÑñÁáÉéÍíÓóÚú]+['\-]{0,1}[A-Za-zÑñÁáÉéÍíÓóÚú]+))*$/.test(nombre); if (nombreTest === false) { alert("Ingrese su primer nombre correctamente"); return false; }
```

- `^`: Con esto se inicia la cadena
- `[A-Za-zÑñÁáÉéÍíÓóÚú]` Permite la incorporación de las letras mayúsculas y minúsculas que estén en el abecedario, así como también las Ñ y distintas vocales con acentos.
- `+` Permite la información incorporada pueda tener más de una palabra dentro del campo (en este caso, el campo nombre) separadas por un espacio
- `['\-]` Esta expresión permite incorporar apóstrofes y guiones al nombre
- `{0,1}` Esta expresión permite las veces que puede repetirse un elemento o sea 0 o 1 vez
- `(nombreTest === false)` Esta expresión busca activar la alerta pertinente en caso de no cumplirse las condiciones previamente mencionadas

- $: Indica el final de la cadena.

Dirección:

```
var direccionTest = /^[a-zA-Z0-9À-ÿ\s\-,.#]+$/i.test(direccion);
```

- ✔ `[a-zA-Z0-9À-ÿ\s\-,.#]+` es una expresión que puntualiza cómo espera que se ingrese una dirección teniendo caracteres diferentes al campo antes expuesto para el campo “nombre”. Permitiendo números y caracteres especiales como el numeral entre otros
- ✔ `À-ÿ` Permite la información incorporada pueda tener palabras acentuadas en otros idiomas
- ✔ `\s` Esta expresión permite espacio y tabulaciones
- ✔ `\-,.#` Esta expresión permite la incorporación de guión, punto, cómo y numeral en el campo
- ✔ `$/i` Esta expresión busca cerrarla expresión regular pero la i buscar hacerla insensible a las letras mayúsculas y minúsculas.

Teléfono:

```
var telefonoTest = /^\d{7,}$/g.test(telefono);
```

- ✔ `\d` esta expresión permite cualquier dígito numérico del 0 al 9
- ✔ `{7,}` es una cuantificación de que al menos debe haber 7 número o más ingresados
- ✔ `g` que no se detenga en la primera coincidencia

Usuario:

```
var UsuarioTest = /^[a-zA-Z0-9_-]{4,20}$/.test(Usuario);
```

- ✔ `[a-zA-Z0-9_-]` Esta expresión permite cualquier combinación con letra mayúscula o minúscula del abecedario para conformar el nombre del usuario, así como también Guión bajo, guión del medio y números,
- ✔ `{4,20}` Es una cuantificación restringe a que el nombre de usuario que se debe generar debe tener entre 4 y 20 letras de los valores antes mencionados

Contraseña:

```
var ContraseñaTest =
/^(?=.*\d)(?=.*[a-z])(?=.*[A-Z])(?=.*[\W_]).{8,20}$/.test(Contraseña);
```

- ✔ `(?=.*\d)`Es un patrón de búsqueda que si hay algún dígito y el asterisco permite cualquier carácter antes del dígito.
- ✔ `(?=.*[a-z])`Es patrón busca las letras minúsculas del abecedario en la contraseña
- ✔ `(?=.*[A-Z])`Esta expresión es igual a la anterior pero con las letras mayúsculas.
- ✔ `(?=.*[\W_]`En este caso se busca verificar si hay un carácter especial o subrayado en la cadena.

Email:

```
var mailTest = /^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/.test(mail);
```

- ✔ `@` Al poner en la expresión después del (+) el arroba se busca puntualizar una separación donde este carácter está en medio de las palabras ingresadas
- ✔ `\.` Aquí se busca que igual al caso anterior con el arroba la separación entre ambas palabras debe darse con un punto en el medio, ambos caracteres deben estar de manera obligatoria en el dato ingresado para la validación positiva del campo.
- ✔ `[a-zA-Z]{2,}` Aquí se pone en claro que el dominio debe tener al menos 2 letras, generalmente las páginas oficiales de los países tienden a tener esa característica “AR” “RU” “ES”, etc.

- ✔ En definitiva las condiciones en esta expresión buscan condicionar la formación de la contraseña ingresada por el usuario a las siguientes características:
  - ✔ `Contiene al menor un digito`
  - ✔ `Contiene al menos una letra minúscula`
  - ✔ `Contiene al menos una letra mayúscula`

✔ `Contiene al menos un carácter especial o subrayado`
✔ `Tiene una longitud entre 8 y 20 caracteres.`

RESUMEN Y CONCLUSIÓN

En este documento, hemos presentado una visión completa de Randart, una plataforma en línea diseñada para conectar artistas, coleccionistas y amantes del arte en un entorno creativo y cultural. Desde el objetivo principal hasta la metodología de desarrollo y la estructura del sitio, hemos detallado los aspectos clave de este emocionante proyecto.

El proyecto Randart se ha desarrollado con un enfoque ágil que permitió la adaptabilidad y la incorporación de nuevas ideas a medida que avanzábamos en su construcción. La colaboración efectiva de un equipo diverso ha sido fundamental para el éxito de este proyecto. Las reuniones regulares, la distribución de tareas y el uso de herramientas como GitHub han permitido mantener un flujo de trabajo eficiente y asegurar la calidad del resultado final.

La estructura del sitio, con sus páginas principales, ha sido diseñada con un estilo minimalista que destaca la esencia del arte. La utilización de fetch para cargar dinámicamente el contenido de la galería demuestra una técnica avanzada para mantener la plataforma actualizada y atractiva.

En resumen, Randart no solo busca proporcionar un espacio para el arte en línea, sino que también busca fomentar la creatividad y la cultura artística. Esperamos que esta plataforma sea un lugar de encuentro para artistas y amantes del arte, donde las obras puedan ser apreciadas y compartidas de manera aleatoria, enriqueciendo la experiencia artística de todos los involucrados.

A medida que el proyecto Randart avanza y se lanza al público, esperamos que siga creciendo y evolucionando, cumpliendo con su objetivo de conectar a las personas a través del arte y la creatividad.

Gracias por ser parte de esta emocionante travesía artística con nosotros.

Si tienes alguna pregunta adicional o necesitas más información, no dudes en contactarnos. ***¡Bienvenido a Randart!***

Contactos:

- Ruiz Mattar, Nahuel - nrmattar@gmail.com
- Esposito, Maximiliano Roberto - maximilianoroberto.esposito@gmail.com
- López Becerra, Analía - analialopezbecerra@live.com.ar
- Ponce, Emanuel David - emanuelponce18@gmail.com