Propuestas de proyecto final para la comisión de JavaScript

En primera instancia, recordando los criterios formales relacionados a los requisitos de presentación del proyecto:

1) Tu trabajo debe ser personal y original y no presentar copia o plagio ni total o parcial de otros trabajos de Coderhouse o de la web. En el caso de haber plagio total, el trabajo no será corregido ni aprobado. En caso de haber una copia parcial en alguna de las partes del trabajo, se te pedirá rehacer dicha parte.
2) El trabajo debe ser entregado con permisos de comentador para que el docente pueda acceder a su visualización. En caso de entregar un link privado o vacío, se debe solicitar al estudiante que entregue con permisos de accesos de comentador para que su trabajo pueda ser corregido.
3) El trabajo es presentado bajo el formato solicitado (rar o preferentemente github). En caso de no entregar en el formato solicitado, se debe pedir al estudiante la entrega bajo el formato establecido para la corrección de su trabajo.

Con respecto a los criterios de evaluación, se corregirá el código teniendo en cuenta los siguientes aspectos y requisitos:

Criterios	Descripción	Alcance	Puntaje parcial
Funcionalidad	Se simula uno o más flujo de trabajo en termino de entrada-procesamiento-salida y no se advierten errores de cómputo.	Óptimo	2
Interactividad	Se capturan entradas empleando inputs y eventos adecuados. Las salidas son coherentes en relación a los datos ingresados y se visualizan en el HTML de forma asíncrona.	Óptimo	2
Escalabilidad	Se declaran funciones con parámetros para definir instrucciones con una tarea específica.Se definen objetos con propiedades y métodos relevantes al contexto. Se emplean arrays para agrupar valores y objetos de forma dinámica. El recorrido de las colecciones es óptimo.	Óptimo	2
Integridad	Se definen el código JavaScript en un archivo .js, referenciándolo correctamente desde el HTML. La información estática en formato JSON se utiliza adecuadamente, cargandose de forma asíncrona.	Óptimo	2
Legibilidad	Los nombres de variables. funciones y objetos son significativos para el contexto, las instrucciones se escriben de forma legible y se emplean comentarios oportunos. El código fuente es ordenado en términos de declaración y secuencia.	Óptimo	2

Cantidad de Criterios	5	1		Nota final	Nivel obtenido
				7 - 10	Aprobado
Criterios		**Descripción**		0 - 7	Desaprobado
Funcionalidad	ÓPTIMO	Se simula uno o más flujo de trabajo en termino de entrada-procesamiento-salida y no se advierten errores de cómputo.		Alcance por aspecto	Puntaje asignado por alcance
	CORRECTO	Se simula uno o más flujos de trabajo en terminos de entrada-proceso-salida.		Óptimo	2
	BAJO	Se simula al menos uno, o ningún, flujo de trabajo en terminos de entrada, proceso y salida.		Correcto	1
	NULO	No es posible corregir este criterio pues no se encuentra en el Proyecto Final.		Bajo	0.5
Interactividad	ÓPTIMO	Se capturan entradas empleando inputs y eventos adecuados. Las salidas son coherentes en relación a los datos ingresados y se visualizan en el HTML de forma asíncrona.		Nulo	0
	CORRECTO	Se capturan entradas empleando inputs y eventos adecuados. Las salidas son coherentes en relación a los datos ingresados y se visualizan en el HTML.			
	BAJO	Las entradas y salidas capturadas no son coherentes a los datos ingresados. ó... No se capturan entradas ni se efectúan salidas de datos coherentes.			
	NULO	No es posible corregir este criterio pues no se encuentra en el Proyecto Final.			
Escalabilidad	ÓPTIMO	Se declaran funciones con parámetros para definir instrucciones con una tarea específica.Se definen objetos con propiedades y métodos relevantes al contexto. Se emplean arrays para agrupar valores y objetos de forma dinámica. El recorrido de las colecciones es óptimo.			
	CORRECTO	Se declaran funciones con parámetros para definir instrucciones con una tarea específica.Se definen objetos con propiedades y métodos relevantes al contexto. Se emplean arrays para agrupar valores y objetos relacionados.			
	BAJO	Se establece separación funcional de las instrucciones. Se emplean estructuras de datos para definir y agrupar datos símiles.			
	NULO	No es posible corregir este criterio pues no se encuentra en el Proyecto Final.			
Integridad	ÓPTIMO	Se definen el código JavaScript en un archivo .js, referenciándolo correctamente desde el HTML. La información estática en formato JSON se utiliza adecuadamente, cargandose de forma asíncrona.			
	CORRECTO	Se definen el código JavaScript en un archivo .js, referenciándolo correctamente desde el HTML. La información estática en formato JSON se utiliza adecuadamente.			
	BAJO	Se definen el código JavaScript dentro del documento HTML			
	NULO	No es posible corregir este criterio pues no se encuentra en el Proyecto Final.			
Legibilidad	ÓPTIMO	Los nombres de variables. funciones y objetos son significativos para el contexto, las instrucciones se escriben de forma legible y se emplean comentarios oportunos. El código fuente es ordenado en términos de declaración y secuencia.			
	CORRECTO	Los nombres de variables. funciones y objetos son significativos para el contexto, las instrucciones se escriben de forma legible y se emplean comentarios oportunos.			
	BAJO	Los nombres de variables. funciones y objetos son significativos para el contexto, pero se presentan diversos puntos de mejora			
	NULO	No es posible corregir este criterio pues no se encuentra en el Proyecto Final.			

El proyecto final debe incluir y relacionar cada uno de los contenidos vistos y trabajados en clase. Se precisará que los estudiantes desarrollen una web y que esta web sea dinámica debido a la implementación del lenguaje JavaScript. La lista de temas a incluir en el desarrollo del proyecto debe ser la siguiente:

1) Variables
2) Datos primitivos y datos objeto: string, number, boolean, arrays y objetos.
3) Operadores (aritméticos y condicionales)
4) Estructuras condicionales.
5) Bucles o métodos de arrays.
6) Funciones
7) Storage: localStorage y/o sessionStorage
8) DOM
9) Eventos
10) Librerías
11) Fetch: API o JSON

El estudiante, si lo desea, puede implementar código que exceda lo detallado y visto en clases, siempre y cuando se cumplan con los requisitos, ya que se evaluará en torno a ellos. **También puede utilizar frameworks de CSS para el armado de la web: Bulma, Boostrap, Tailwind, etc.**

CASO ECOMMERCE

Se desarrollará una web dinámica, que interactúe con el usuario a través de diferentes medios:

- **Modo oscuro**: El usuario debe poder elegir si, al navegar en la web, el modo en el cual se visualiza la interfaz sea en claro u oscuro. Este modo tiene que otorgar la posibilidad de que perdure en el tiempo, es decir, una vez que el usuario defina su preferencia, el modo debe persistir a un cierre de pestaña o navegador. El botón que desencadene el evento, debe hallarse visible en la barra de navegación, desde todos los apartados de la página web.
- La **barra de navegación** debe poder permitir movilidad y ser vista desde todos los apartados. El traslado entre los apartados puede darse a través de diferentes archivos HTML o bien con la habilitación/deshabilitación de la visibilidad de los contenedores en donde se desarrollan. Esta nav debe permitirle al usuario poder desplazarse entre las siguientes secciones: **HOME, PRODUCTOS, CARRITO Y CONTACTO**.
- Se precisará un footer que tenga información de contacto, la cual puede ser falsa. Este pie de página debe ser visualizado desde todos los apartados de la web.
- **SECCIÓN HOME**: Se debe poder visualizar un Carrousel, el cual puede ser implementado a través de librerías. Como sugerencia Swiper. Este elemento debe mostrar imágenes relacionadas al contenido de la web. A su vez, el home debe visualizar productos en oferta y que los mismos puedan ser añadidos al carrito desde allí.
- **SECCIÓN PRODUCTOS**: El usuario debe poder visualizar todos los productos que ofrezca la tienda, estos productos no pueden estar maquetados desde HTML, deben realizarse con el método fetch y además incluir los nodos de los productos desde js. A su vez, la web debe darle la posibilidad al usuario de que ordene estos productos de manera ascendente (A-Z) o descendente (Z-A) y que este comportamiendo genere que los cambios se apliquen en el instante en el cual se desencadeno el evento. También se pueden filtrar y visualizar por oferta. Estos productos deben poder añadirse al carrito desde un botón, que se debe encontrar incluido en la tarjeta del producto.
- **SECCIÓN CARRITO**: Los productos añadidos desde **HOME** o **PRODUCTOS**, deben poder visualizarse en este apartado en forma de lista (no necesariamente se debe hacer una tabla), a través de JS. A su vez se debe permitir que el usuario elimine un elemento del carrito o elimine todos los elementos del carrito. En esta sección, nos concentraremos en el cierre de la venta, por lo tanto debemos tener en cuenta de que una vez de que el usuario terminó con el proceso, los productos no deben persistir en el carrito de compras. Se debe, mediante un evento, enviar una notificación al usuario que exprese la finalización de dicho transcurso.
- **SECCIÓN CONTACTO**: En este apartado, el usuario tendrá la posibilidad de poder llenar un formulario, que permita contactarse con la empresa. Este formulario tiene que validar que los datos que se ingresar son correctos. Ej: el campo que reciba un mail, debe cotejar que el dato ingresado contenga una @, distintiva de este tipo de recurso, que un campo nombre debe tener más de 2 caracteres, etc.

CASO BLOG

Se desarrollará una web dinámica, que interactúe con el usuario a través de diferentes medios:

- Se precisará de una barra de navegación que permita al usuario navegar por los siguientes apartados: HOME, FAVORITOS, INICIAR SESIÓN. En el caso de que el usuario haga click a FAVORITOS y no se encuentre logeado, se debe redirigirlo a INICIAR SESIÓN. Esta barra de navegación debe ser visible desde todas las secciones.
- Se debe integrar un footer que tenga información falsa del sitio. Este pié de página debe ser visualizado desde todos los apartados.
- **Modo oscuro**: El usuario debe poder elegir si, al navegar en la web, el modo en el cual se visualiza la interfaz sea claro u oscuro. Este modo tiene que otorgar la posibilidad de que perdure en el tiempo, es decir, una vez que el usuario defina su preferencia, el modo debe persistir a un cierre de pestaña o navegador. El botón que desencadene el evento, debe hallarse visible en la barra de navegación, desde todos los apartados de la página web.
- **HOME**: Esta sección debe contar con al menos 20 posteos. Estos posteos deben generarse desde JS, los nodos que se incluyan al DOM deben instanciarse desde un FETCH. Cada "publicación" debe contar con los siguientes datos: nombreDeUsuario, post, img (en el caso de que tenga una, la dirección de la imagen, sino un false como valor), fecha y hora. Cada posteo debe permitir al usuario darle like a esa publicación a través de un botón integrado en la tarjeta. Se debe enviar una notificación al usuario, cada vea que añada a favoritos un post.
- **FAVORITOS**: Esta sección debe contener la visualización de todos los posteos favoritos del usuario. Las publicaciones likeadas deben ser visualizadas desde este apartado. El usuario debe tener la posibilidad de eliminar una publicación, que ya no quiera visualizar o eliminar todas las publicaciones. Estas publicaciones deben persistir en el tiempo, es decir, al cerrar la pestaña o el navegador, si el usuario vuelve acceder a la web, debe seguir viendo lo que envió a favoritos. Deben poder ordenarse las publicaciones de manera ascendente (A-Z) y descendentes (Z-A), en relación a su fecha.
- **INICIAR SESIÓN**: Este apartado solo será visualizado, siempre y cuando el usuario no haya logeado su sesión. Contará con un formulario que permita al usuario ingresar sus datos, de ser incorrectos, se debe notificar en el formulario que el logeo fue incorrecto. En caso contrario, se deberá modificar la barra de navegación, para que la visibilidad de este apartado no sea posible. A su vez, la barra de navegación debe habilitar al usuario la posibilidad de desloggearse de la web.

WEB INTERACTIVA (API)

Se desarrollará una web dinámica, que interactúe con el usuario a través de diferentes medios:

- Se precisará de una barra de navegación que tenga los siguientes apartados: **HOME, BUSCAR, FAVORITOS Y CONTACTO.** El usuario debe tener la posibilidad de trasladarse entre secciones.
- Se debe integrar un footer que visibilice información de redes sociales del creador de la web. Este pié de página debe ser visualizado desde todos los apartados.
- **Modo oscuro**: El usuario debe poder elegir si, al navegar en la web, el modo en el cual se visualiza la interfaz sea claro u oscuro. Este modo tiene que otorgar la posibilidad de que perdure en el tiempo, es decir, una vez que el usuario defina su preferencia, el modo debe persistir a un cierre de pestaña o navegador. El botón que desencadene el evento, debe hallarse visible en la barra de navegación, desde todos los apartados de la página web.
- **HOME:** En este apartado, debe visualizarse un header que represente la identidad o contenido de la web. En el cuerpo de la página, a través de una API, deben visualizarse hasta 20 tarjetas. Todas las tarjetas deben estar generadas desde JS.
- **BUSCAR:** En esta sección, se deberá tener la posibilidad de buscar un elemento/personaje. Esta búsqueda debe visualizar los resultados encontrados. El usuario debe tener la posibilidad de añadir a favoritos, mediante un botón hallado en la tarjeta, a los elementos seleccionados. Estos elementos deben persistir a un cierre de ventana o navegador. El usuario debe tener la posibilidad de quitar de favoritos a una de las tarjetas o a todas, mediante eventos. Cada vez que se añada a favoritos una tarjeta, el usuario debe recibir una notificación de su interacción.
- **FAVORITOS**: Esta sección debe contener la visualización de todos las tarjetas favoritas del usuario. Los personajes/elementos favoritos deben ser visualizadas desde este apartado. El usuario debe tener la posibilidad de eliminar una tarjeta, que ya no quiera visualizar o eliminar todas las tarjetas. Estas publicaciones deben persistir en el tiempo, es decir, al cerrar la pestaña o el navegador, si el usuario vuelve acceder a la web, debe seguir viendo lo que envió a favoritos. Deben poder ordenarse las publicaciones de manera ascendente (A-Z) y descendentes (Z-A), en relación a su nombre.
- **SECCIÓN CONTACTO**: En este apartado, el usuario tendrá la posibilidad de poder llenar un formulario, que permita contactarse con la creador de la web. Este formulario tiene que validar que los datos que se ingresar son correctos. Ej: el campo que reciba un mail, debe cotejar que el dato ingresado contenga una @, distintiva de este tipo de recurso, que un campo nombre debe tener más de 2 caracteres, etc.

Estos puntos planteados y a desarrollar, cumplen con todos los criterios de evaluación y sirven como puntapié para el desarrollo de una web consistente que utilice todos los recursos impartidos durante la cursada. En el caso de que el estudiante no llegue a completarlos, no significa que vaya a reprobar. Sin embargo se seguirán evaluando de todos modos que se hayan integrado todos los elementos vistos en clase.