Que debe hacer  
  
  
  
  
FEED

 **Carga de publicaciones:**

* Cuando se abre la página del feed, se envía una solicitud al servidor para obtener todas las publicaciones que otros usuarios han realizado.
* Estas publicaciones se muestran en la pantalla, con la información del usuario que la publicó, la fecha de publicación, y el contenido de la publicación.
* Si la publicación incluye un archivo adjunto, este se mostrará automáticamente. Si es una imagen, se verá directamente; si es otro tipo de archivo, aparecerá como un enlace para que el usuario lo descargue o lo abra.

 **Comentarios en publicaciones:**

* Debajo de cada publicación, hay un botón de comentarios. Al hacer clic en este botón, se despliega o se oculta el cuadro de comentarios.
* El cuadro de comentarios muestra los comentarios anteriores que otros usuarios han dejado, y permite que el usuario actual escriba y envíe su propio comentario.
* Cuando el usuario envía un comentario, este se guarda en el servidor y se actualiza la lista de comentarios de esa publicación.

 **Reacciones a publicaciones:**

* Para cada publicación, el usuario puede ver si ya ha reaccionado (por ejemplo, con un "Me Gusta" o "Me Encanta") o no ha reaccionado.
* Si ya reaccionó, el feed muestra la reacción del usuario como un botón con un emoji y el nombre de la reacción (por ejemplo, 👍 Me Gusta).
* Si no ha reaccionado, aparece un botón para "Reaccionar". Al hacer clic en este botón, se despliegan varias opciones de reacciones como "Me Gusta", "Me Encanta", "Me Sorprende", "Me Enoja", "Me Entristece", cada una representada con un emoji.
* Cuando el usuario selecciona una reacción, esta se guarda en el servidor y se actualiza en la pantalla para reflejar la nueva reacción.

 **Actualización dinámica:**

* Cada vez que el usuario publica un comentario o cambia su reacción a una publicación, el feed se actualiza automáticamente sin necesidad de recargar la página.
* También, cada vez que se abre el cuadro de comentarios, se cargan los comentarios más recientes desde el servidor, para asegurarse de que el usuario siempre vea la información más actualizada.

HISTORIASSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS

Te explico lo que hace este código paso a paso:

### 1. \*\*Formulario de subida de historias:\*\*

- \*\*Mostrar/Ocultar formulario:\*\* El código tiene una función para mostrar o ocultar el formulario donde los usuarios pueden subir historias. Si el formulario no está visible, se muestra cuando se llama a la función, y si está visible, se oculta.

- \*\*Subir historia:\*\*

- Al enviar el formulario, se toma el archivo y el contenido ingresado por el usuario.

- Se envía esta información al servidor con una petición `POST`.

- Cuando el servidor responde, si la historia se sube correctamente, se muestra un mensaje al usuario y el formulario se oculta automáticamente.

- Después de subir la historia, se vuelve a cargar la lista de historias para que la nueva historia aparezca en el feed.

### 2. \*\*Carga de historias:\*\*

- \*\*Obtener historias desde el servidor:\*\* La función `loadHistorias` hace una solicitud al servidor para obtener las historias publicadas por los usuarios.

- \*\*Mostrar historias en pantalla:\*\*

- Por cada historia recibida, se crea un contenedor donde se muestra el nombre del usuario que la subió, y el contenido de la historia.

- Si la historia contiene un archivo multimedia (imagen o video), este se mostrará también. Los videos incluyen controles para reproducir, pausar, etc.

- Las imágenes y videos se pueden expandir a pantalla completa al hacer clic en ellos.

### 3. \*\*Partículas flotantes en el fondo:\*\*

- \*\*Canvas para partículas:\*\* Se usa un elemento `canvas` para crear un efecto visual de partículas flotantes en el fondo de la página.

- \*\*Ajuste dinámico del tamaño:\*\* El tamaño del canvas se ajusta automáticamente al tamaño de la ventana del navegador.

- \*\*Partículas:\*\*

- Se crean múltiples partículas (hasta 100) con propiedades aleatorias como tamaño, posición y dirección de movimiento.

- Las partículas se mueven en diferentes direcciones dentro del canvas y cambian de dirección cuando tocan los bordes.

- \*\*Animación:\*\* Cada partícula se actualiza constantemente para crear un efecto de movimiento continuo, y el proceso se repite indefinidamente con la función `requestAnimationFrame` que asegura una animación fluida.

- \*\*Ajuste a la pantalla:\*\* Si el tamaño de la ventana cambia, el canvas y las partículas se reajustan para seguir ocupando todo el espacio visible.

En resumen, este código permite a los usuarios subir y ver historias (imágenes y videos), y añade un efecto visual de partículas en movimiento en el fondo para mejorar la experiencia del usuario en la página.

RETOSSSSSSSSSSSSSSSSSS

Aquí tienes la descripción de lo que hace este código:

### 1. \*\*Mostrar/Ocultar el formulario para subir retos:\*\*

- Al hacer clic en el botón con ID `nuevoRetoButton`, se alterna entre mostrar y ocultar el formulario para subir un reto. Si el formulario está oculto, se muestra, y si está visible, se oculta.

### 2. \*\*Subida de retos:\*\*

- Al enviar el formulario de subida de retos, se detiene el comportamiento por defecto (recargar la página), y en su lugar, se envía una solicitud `POST` al servidor con los datos del reto mediante `FormData`.

- El servidor responde con un mensaje. Si la subida es exitosa, se muestra una alerta y se recargan los retos visibles en la página.

### 3. \*\*Carga de retos:\*\*

- La función `loadRetos` hace una solicitud al servidor para obtener todos los retos publicados.

- Por cada reto recibido:

- Se muestra el nombre del usuario que publicó el reto, el contenido del reto y, si existe, una imagen o archivo adjunto (con un botón para descargarlo si no es una imagen).

- Se muestra la fecha de publicación.

- Se genera un botón que permite ver o ocultar las respuestas asociadas al reto.

- También hay un botón para mostrar u ocultar el formulario para subir una nueva respuesta al reto.

- El formulario de respuesta incluye campos para el contenido y un archivo opcional.

- Después de mostrar cada reto, la función `loadRespuestas` se llama automáticamente para cargar las respuestas del reto específico.

### 4. \*\*Mostrar/Ocultar respuestas de un reto:\*\*

- Cada reto tiene un botón para ver las respuestas. Al hacer clic en este botón, se alterna entre mostrar y ocultar el contenedor de respuestas asociadas al reto.

### 5. \*\*Subida de respuestas:\*\*

- Cada reto tiene un formulario de respuestas asociado. Cuando se envía este formulario, se manda una solicitud `POST` al servidor con el contenido de la respuesta y el archivo (si se adjunta alguno).

- Si la respuesta se sube correctamente, se recargan las respuestas para que la nueva respuesta aparezca automáticamente.

### 6. \*\*Carga de respuestas:\*\*

- La función `loadRespuestas` solicita al servidor todas las respuestas de un reto específico usando su ID.

- Cada respuesta muestra el nombre del usuario que la publicó, el contenido y, si tiene un archivo adjunto, se mostrará una imagen, video o un enlace para descargar el archivo según su tipo.

- La fecha de publicación también se muestra.

### 7. \*\*Renderizado de archivos adjuntos:\*\*

- La función `renderFile` se encarga de decidir cómo mostrar un archivo adjunto dependiendo de su tipo:

- Imágenes (`jpg`, `png`, `gif`) se muestran directamente como una imagen.

- Videos (`mp4`, `webm`, `ogg`) se muestran con un reproductor de video.

- Otros archivos se muestran como un enlace para descargarlos.

### 8. \*\*Carga inicial de retos:\*\*

- Cuando la página se carga por primera vez, se llama automáticamente a `loadRetos` para mostrar todos los retos disponibles.

En resumen, este código permite a los usuarios:

- Publicar retos con texto e imágenes/archivos adjuntos.

- Ver los retos publicados por otros usuarios.

- Publicar respuestas a los retos, también con texto e imágenes/archivos adjuntos.

- Ver las respuestas de cada reto, con opciones para mostrar/ocultar tanto las respuestas como los formularios de respuesta.

INICIOOOOOOOOOOOOOOOOOOOOOOOOOOO

Este código es una página sencilla de \*\*subida de publicaciones\*\* con un formulario. A continuación te explico su funcionamiento:

### 1. \*\*Estructura HTML:\*\*

- El formulario tiene un campo de texto (`textarea`) para que los usuarios escriban el contenido de su publicación y un campo de archivo (`input type="file"`) para permitir la subida de archivos (como imágenes o videos).

- El botón "Subir" envía los datos del formulario.

- El botón "Cancelar" redirige al usuario al \*\*feed\*\* sin enviar nada.

- Hay un párrafo (`<p id="message"></p>`) que mostrará mensajes de éxito o error después de que el usuario intente subir algo.

### 2. \*\*Manejador del formulario:\*\*

- Cuando el formulario es enviado (clic en "Subir"), la función escucha el evento `submit` y lo \*\*previene\*\* para evitar que se recargue la página de forma tradicional.

- Los datos del formulario se empaquetan en un objeto `FormData`, que es ideal para manejar tanto texto como archivos.

- Una vez preparado el `FormData`, se hace una solicitud `POST` a la ruta `/upload` para subir los datos al servidor.

- Dependiendo de la respuesta del servidor:

- Si la subida es exitosa, se muestra un mensaje de éxito en color verde y el formulario se limpia.

- Si hubo un error, se muestra un mensaje en rojo.

- Después de un segundo tras el éxito, se redirige automáticamente al \*\*feed\*\* (página `feed.html`).

### 3. \*\*Manejador del botón "Cancelar":\*\*

- Si el usuario hace clic en "Cancelar", será redirigido directamente al \*\*feed\*\* (`feed.html`) sin que se suba ningún dato o archivo.

### 4. \*\*Manejo de errores:\*\*

- Si ocurre un error durante la solicitud (por ejemplo, un fallo de red), se captura en el bloque `catch` y se muestra un mensaje informando que hubo un problema al subir la publicación.

En resumen, esta página permite a los usuarios:

- Escribir una publicación con texto y un archivo opcional.

- Subir la publicación al servidor.

- Cancelar la operación y volver al \*\*feed\*\* en cualquier momento.

- Ver mensajes claros de éxito o error después de la subida.