Задание: Скринсейвер с котами и собаками.

Нужно реализовать скринсейвер на JavaScript, который каждые 3 секунды подгружает и показывает новое случайное изображение животного с внешнего API.

Требования:

- Использовать асинхронные запросы (fetch + async/await) для загрузки картинок.
- Реализовать переключатель "Коты / собаки ":
- Коты загружаются с TheCatAPI
- Собаки загружаются с Dog CEO API
- Сделать кнопку "Пауза", которая останавливает автоматическую смену изображений. При повторном нажатии — возобновляет.
- Каждое новое изображение должно полностью загружаться, прежде чем будет показано пользователю.
- При ошибке загрузки нужно выводить сообщение "Не удалось загрузить изображение, попробуйте снова".

Дополнительно (по желанию):

- Добавить плавное исчезновение/появление картинок (через CSS transition или JS).

Шпаргалка по работе с API и картинками (JS)

1. fetch — получение данных с сервера const response = await fetch("https://example.com/api"); Делает HTTP-запрос по указанному адресу.

Возвращает объект Response.

2. Проверка успешности ответа

```
if (!response.ok) {
  throw new Error("Ошибка при загрузке данных");
}
```

response.ok → true, если код ответа от сервера 200–299. Если ошибка → можно вывести сообщение пользователю.

3. Чтение ответа в формате JSON const data = await response.json();
Метод .json() преобразует тело ответа в объект JS.
Работает асинхронно, поэтому нужен await.
Примеры API:
Cat API → data[0].url
Dog API → data.message

4. async/await async перед функцией → внутри можно использовать await. await ждёт завершения асинхронной операции (например, загрузки данных).

```
async function getData() {
  const response = await fetch("https://api...");
const data = await response.json();
console.log(data);
}
```

5. Работа с <img> и загрузкой картинок

```
const img = document.getElementById("animal-image");
// меняем картинку
img.src = "https://example.com/cat.jpg";

// ждём, пока загрузится img.onload
= () => {
  console.log("Картинка готова к показу!");
};

// если ошибка img.onerror
= () => {
  console.log("Ошибка при загрузке картинки");
```

```
};
```

6. Плавное появление картинки (CSS + JS)
```
img {   opacity: 0;
  transition: opacity 0.8s ease;
}

img.onload = () => {
  img.style.opacity = 1; // плавно проявляется
};
img.style.opacity = 0;   // скрываем перед подменой
```

7. Таймеры setInterval(fn, время) — вызывает функцию каждые N миллисекунд.
clearInterval(id) — останавливает интервал. setTimeout(fn, время) — запускает один раз через N миллисекунд.

8. Обработка ошибок
```
try {
  const response = await fetch("https://...");
  if (!response.ok) throw new Error("Ошибка ответа сервера");
  const data = await response.json();
} catch (error) {
  console.error("Ошибка:", error);
}
```

Шпаргалка по Cat & Dog APIs The Cat API (thecatapi.com)
https://api.thecatapi.com/v1/images/search

Dog CEO API (dog.ceo/dog-api)
https://dog.ceo/api/breeds/image/random

шаги для выполнения задания:
Получить ссылку на картинку с API (котики или пёсики).

Подставить её в img.src.
Использовать img.onload, чтобы плавно показать картинку.
Запускать обновление через setInterval.
Сделать кнопки для паузы и переключения.