**Descrierea**

Aceasta aplicatie implementeaza un sistem de procesare a imaginilor care preia o imagine de la un API public (Dog API), aplica doua efecte (Mirror si Gray-Scale) asupra imaginii si afiseaza timpul necesar pentru fiecare etapa.

**Partea teoretica**

1. **Procesarea imaginilor**

Am folosit:

* **Mirror Effect**: Pixelii din partea stanga a imaginii sunt schimbati cu cei din partea dreapta.
* **Gray-Scale**: Imaginea este convertita in alb-negru prin calcularea mediei valorilor culorilor rosu, verde si albastru (RGB). (

**2. Programare asincrona**

JavaScript permite rularea sarcinilor asincrone (precum preluarea imaginilor de pe internet) fara a bloca executia altor functii.

**3. API Dog**

API-ul utilizat in acest proiect (https://dog.ceo/api/breeds/image/random) ofera imagini cu caini. Raspunsul este in format JSON, continand URL-ul imaginii si un status.

**Descrierea implementarii**

Aplicatia este formata din trei parti: HTML, CSS si JavaScript

**1. HTML**

Codul HTML defineste structura paginii:

* Un buton pentru generarea imaginii
* Un canvas pentru afisarea imaginii procesate
* Sectiuni pentru afisarea informatiilor JSON si a timpilor de procesare

**2. CSS**

Fisierul style.css defineste stilurile paginii: header,canvas,buton,div uri etc.

Culorile predominante sunt:

* **#f4a261**
* **#115d8c**

**3. JavaScript**

Fisierul script.js implementeaza logica aplicatiei:

* **Eveniment pe butonul de generare** :preia imaginea si masoara timpul timpul necesar pentru preluarea si procesarea imaginii
* **Procesarea imaginii in 4 segmente**: fiecare segment este procesat cu o intarziere de 1 secunda

**Descrierea functionala a aplicatiei implementate**

1. **Interfata simpla**:
   * Un buton centralizat genereaza imaginea
   * Imaginea procesata este afisata in canvas
2. **Afisare timpi**:
   * Timpii de procesare pentru Mirror si Gray-Scale sunt afisati intr-o lista
3. **Mesaj de eroare**
   * Daca imaginea nu poate fi preluata, utilizatorul primeste un mesaj de eroare

**Descrierea modulelor**

1. **resetProcessingTimes**:sterge lista cu timpii de procesare afisati anterior.
2. **displayJSON**:afiseaza informatiile JSON obtinute de la API.
3. **processImage**:imparte imaginea in 4 segmente si aplica efectele Mirror si Gray-Scale ; masoara timpul total pentru fiecare efect.
4. **displayTime**:afiseaza timpii masurati pentru etapele de procesare.

**Bibliografie**

**-W3Schools - JavaScript Fetch API**

**-W3Schools - HTML Canvas**