**Лабораторна робота № 11**

**Використання Fullscreen API, Local Storage API, Geolocation API та Visibility API та основи Fetch API***.*

**Хід роботи:**

**Завдання 1:**

Користуючись запитом до АРІ ресурсу<https://dog.ceo/api/breeds/image/random> відобразити на сторінці галерею фотографій собак по натисканню на кнопку.

**Лістинг програми:**

HTML file:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Task 1</title>

    <link rel="stylesheet" href="style.css">

</head>

<body>

    <button onclick="loadDogImages()">Add Photo</button>

    <div id="gallery"></div>

    <script src="script.js"></script>

</body>

</html>

Java Script file:

async function loadDogImages() {

    const gallery = document.getElementById('gallery');

    try {

        const response = await fetch('https://dog.ceo/api/breeds/image/random');

        const data = await response.json();

        console.log('API Response:', data);

        if (response.ok && data.status === 'success' && data.message && data.message.trim() !== '') {

            const imgElement = document.createElement('img');

            imgElement.src = data.message;

            gallery.appendChild(imgElement);

        } else

            console.error('Failed to fetch dog image:', data.message);

    } catch (error) {

        console.error('Error fetching dog image:', error.message);

    }

}

CSS file:

#area {

    position: relative;

    width: 200px;

    height: 200px;

    border: 5px solid yellow;

    outline: 1px solid black;

    display: flex;

    top: 50%;

    left: 50%;

    transform: translate(-50%);

}

.block {

    position: absolute;

    opacity: 0.7;

}

CSS file:

#gallery {

    display: flex;

    flex-wrap: wrap;

}

img {

    width: 250px;

    height: 250px;

    object-fit: cover;

    border: 1px solid #ddd;

}

**Результати виконання:**

****

**Завдання 2:**

Додайте кнопку “Fullscreen View”, при натисканні на яку відбудеться відкриття галереї на повний екран.

Під час входу в повноекранний режим змінюйте фоновий колір або додайте анімацію для покращення візуального відображення.

При відображенні галереї у повноекранному режимі повинна з’являтися кнопка для виходу з повноекранного режиму.

**Лістинг програми:**

HTML file:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Task 2</title>

    <link rel="stylesheet" href="style.css">

</head>

<body>

    <button id="fullscreenButton" onclick="toggleFullscreen()">Fullscreen View</button>

    <button id="loadMoreButton" onclick="loadMoreImages()">Load More Dog Images</button>

    <div id="galleryContainer">

        <div id="gallery"></div>

    </div>

    <script src="script.js"></script>

</body>

</html>

Java Script file:

let page = 1;

async function loadDogImages() {

    const gallery = document.getElementById('gallery');

    try {

        const response = await fetch(`https://dog.ceo/api/breeds/image/random/9?page=${page}`);

        const data = await response.json();

        console.log('API Response:', data);

        if (response.ok && data.status === 'success' && data.message && data.message.length > 0) {

            data.message.forEach((imageUrl) => {

                const imgElement = document.createElement('img');

                imgElement.src = imageUrl;

                gallery.appendChild(imgElement);

                imgElement.addEventListener('click', () => enterFullscreen(imgElement));

            });

        } else {

            console.error('Failed to fetch dog images:', data.message);

        }

    } catch (error) {

        console.error('Error fetching dog images:', error.message);

    }

}

function enterFullscreen(element) {

    if (element.requestFullscreen) {

        element.requestFullscreen();

    } else if (element.mozRequestFullScreen) {

        element.mozRequestFullScreen();

    } else if (element.webkitRequestFullscreen) {

        element.webkitRequestFullscreen();

    } else if (element.msRequestFullscreen) {

        element.msRequestFullscreen();

    }

}

function exitFullscreen() {

    if (document.exitFullscreen) {

        document.exitFullscreen();

    } else if (document.mozCancelFullScreen) {

        document.mozCancelFullScreen();

    } else if (document.webkitExitFullscreen) {

        document.webkitExitFullscreen();

    } else if (document.msExitFullscreen) {

        document.msExitFullscreen();

    }

}

function toggleFullscreen() {

    const body = document.body;

    const galleryContainer = document.getElementById('galleryContainer');

    const fullscreenButton = document.getElementById('fullscreenButton');

    if (!document.fullscreenElement) {

        galleryContainer.requestFullscreen();

        body.style.backgroundColor = '#222';

        fullscreenButton.classList.add('exit');

        fullscreenButton.innerText = 'Exit Fullscreen';

    } else {

        exitFullscreen();

        body.style.backgroundColor = '#f4f4f4';

        fullscreenButton.classList.remove('exit');

        fullscreenButton.innerText = 'Fullscreen View';

    }

}

document.addEventListener('fullscreenchange', () => {

    const body = document.body;

    const fullscreenButton = document.getElementById('fullscreenButton');

    if (!document.fullscreenElement) {

        body.style.backgroundColor = '#f4f4f4';

        fullscreenButton.classList.remove('exit');

        fullscreenButton.innerText = 'Fullscreen View';

    }

});

function loadMoreImages() {

    page++;

    loadDogImages();

}

loadDogImages();

CSS file:

body {

    font-family: 'Arial';

    background-color: #ffffff;

    margin: 20px;

    display: flex;

    align-items: center;

    justify-content: center;

    min-height: 50vh;

    flex-direction: column;

    transition: background-color 0.5s;

}

#galleryContainer {

    overflow-y: auto;

    max-height: 100vh;

}

#gallery {

    display: flex;

    flex-wrap: wrap;

    gap: 10px;

}

img {

    width: 100px;

    height: 100px;

    object-fit: cover;

    border: 1px solid #000000;

    border-radius: 20px;

    cursor: pointer;

    transition: transform 0.5s;

    margin-top: 25px;

}

img:hover {

    transform: scale(1.1);

}

#fullscreenButton {

    margin-top: 25px;

    padding: 15px;

    font-size: 16px;

    background-color: #4016fd;

    color: white;

    border: none;

    border-radius: 10px;

    cursor: pointer;

    transition: background-color 0.3s;

}

#fullscreenButton:hover {

    background-color: #45a049;

}

#fullscreenButton.exit {

    background-color: #f44336;

}

#fullscreenButton.exit:hover {

    background-color: #d32f2f;

}

#loadMoreButton {

    margin-top: 25px;

    padding: 5px;

    font-size: 12px;

    background-color: #6708c6;

    color: white;

    border: none;

    border-radius: 10px;

    cursor: pointer;

    transition: background-color 0.3s;

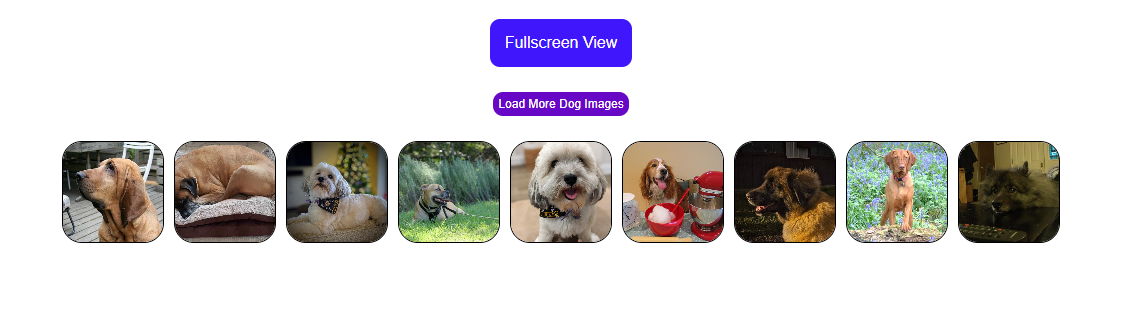
}

#loadMoreButton:hover {

    background-color: #0b7dda;

}

**Результати виконання:**

****

**Завдання 3:**

Реалізуйте збереження стану фотогалереї у Local Storage так, що при закритті сторінки та повторного входу на сторінку відображався останній стан фотогалереї (щоб відображалися ті самі картинки, які були на сторінці перед її закриттям).

**Лістинг програми:**

HTML File:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Task 3</title>

    <link rel="stylesheet" href="style.css">

</head>

<body>

    <button onclick="loadDogImages()">Add Photo</button>

    <div id="gallery" class="flex-container"></div>

    <script src="script.js"></script>

</body>

</html>

JavaScript file:

function getGalleryState() {

    const galleryState = localStorage.getItem('galleryState');

    return galleryState ? JSON.parse(galleryState) : [];

}

function saveGalleryState(images) {

    localStorage.setItem('galleryState', JSON.stringify(images));

}

function displayGalleryState() {

    const gallery = document.getElementById('gallery');

    const galleryState = getGalleryState();

    galleryState.forEach(imageUrl => {

        const imgElement = document.createElement('img');

        imgElement.src = imageUrl;

        gallery.appendChild(imgElement);

    });

}

async function loadDogImages() {

    const gallery = document.getElementById('gallery');

    try {

        const response = await fetch('https://dog.ceo/api/breeds/image/random');

        const data = await response.json();

        console.log('API Response:', data);

        if (response.ok && data.status === 'success' && data.message && data.message.trim() !== '') {

            const imgElement = document.createElement('img');

            imgElement.src = data.message;

            gallery.appendChild(imgElement);

            const galleryState = getGalleryState();

            galleryState.push(data.message);

            saveGalleryState(galleryState);

        } else {

            console.error('Failed to fetch dog image:', data.message);

        }

    } catch (error) {

        console.error('Error fetching dog image:', error.message);

    }

}

displayGalleryState();

CSS file:

#gallery {

    display: flex;

    flex-wrap: wrap;

}

img {

    width: 150px;

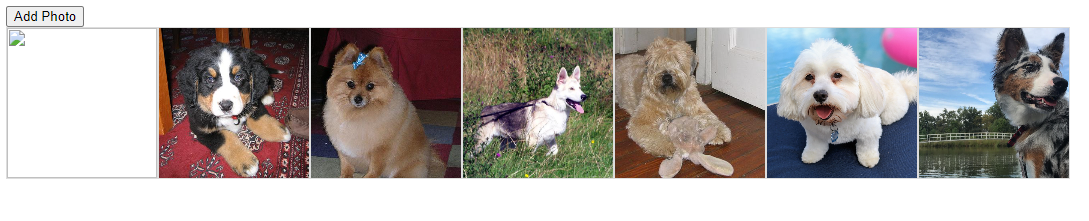
    height: 150px;

    object-fit: cover;

    border: 1px solid #ddd;

}

**Результати виконання:**



**Завдання 4:**

Розмістіть таймер обрахунку часу перебування користувача на сторінці у лівому верхньому куті. При переході до іншої вкладки чи згортанні вікна призупиніть таймер, а при поверненні користувача на сторінці знову запустіть його. Таким чином таймер повинен підраховувати загальний час активного перебування користувача на сторінці (коли сторінка видима для користувача).

**Лістинг програми:**

HTML file:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Task 4</title>

    <link rel="stylesheet" href="style.css">

</head>

<body>

    <div id="timer" class="timer">0:00</div>

    <br><br><br>

    <button onclick="loadDogImages()">Add Photo</button>

    <div id="gallery" class="flex-container"></div>

    <script src="script.js"></script>

</body>

</html>

JavaScript file:

let timerInterval;

let elapsedTime = 0;

function startTimer() {

    timerInterval = setInterval(function () {

        elapsedTime++;

        updateTimerDisplay();

    }, 1000);

}

function updateTimerDisplay() {

    const minutes = Math.floor(elapsedTime / 60);

    const seconds = elapsedTime % 60;

    const formattedTime = `${minutes}:${seconds < 10 ? '0' : ''}${seconds}`;

    document.getElementById('timer').textContent = formattedTime;

}

document.addEventListener('visibilitychange', function () {

    if (document.visibilityState === 'visible') {

        startTimer();

    } else {

        clearInterval(timerInterval);

    }

});

async function loadDogImages() {

    const gallery = document.getElementById('gallery');

    try {

        const response = await fetch('https://dog.ceo/api/breeds/image/random');

        const data = await response.json();

        console.log('API Response:', data);

        if (response.ok && data.status === 'success' && data.message && data.message.trim() !== '') {

            const imgElement = document.createElement('img');

            imgElement.src = data.message;

            gallery.appendChild(imgElement);

        } else {

            console.error('Failed to fetch dog image:', data.message);

        }

    } catch (error) {

        console.error('Error fetching dog image:', error.message);

    }

}

startTimer();

CSS file:

#gallery {

    display: flex;

    flex-wrap: wrap;

}

img {

    width: 150px;

    height: 150px;

    object-fit: cover;

    border: 1px solid #000000;

}

.timer {

    position: fixed;

    top: 10px;

    left: 10px;

    font-size: 18px;

    background-color: #f4f4f4;

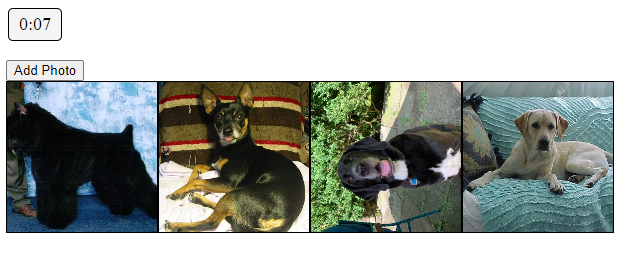
    padding: 5px 10px;

    border: 1px solid #000000;

    border-radius: 5px;

}

**Результати виконання:**

****

**Завдання 5:**

Запитуйте дозвіл на доступ до геолокації користувача. Якщо дозвіл отримано, відображайте у правому верхньому куті сторінки поточне місцезнаходження користувача (широту та довготу).

Оновлюйте інформацію про місцезнаходження при зміні положення користувача.

**Лістинг програми:**

HTML file:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Task 5</title>

    <link rel="stylesheet" href="style.css">

</head>

<body>

    <button onclick="loadDogImages()">Add Photo</button>

    <div id="gallery" class="flex-container"></div>

    <div id="locationInfo" class="location-info"></div>

    <script src="script.js"></script>

</body>

</html>

Java Script file:

async function loadDogImages() {

    const gallery = document.getElementById('gallery');

    try {

        const response = await fetch('https://dog.ceo/api/breeds/image/random');

        const data = await response.json();

        console.log('API Response:', data);

        if (response.ok && data.status === 'success' && data.message && data.message.trim() !== '') {

            const imgElement = document.createElement('img');

            imgElement.src = data.message;

            gallery.appendChild(imgElement);

        } else {

            console.error('Failed to fetch dog image:', data.message);

        }

    } catch (error) {

        console.error('Error fetching dog image:', error.message);

    }

}

function getLocation() {

    if (navigator.geolocation) {

        navigator.geolocation.getCurrentPosition(

            (position) => {

                const locationInfo = document.getElementById('locationInfo');

                const latitude = position.coords.latitude;

                const longitude = position.coords.longitude;

                locationInfo.textContent = `Location: ${latitude}, ${longitude}`;

            },

            (error) => {

                console.error('Error getting location:', error.message);

            }

        );

    } else {

        console.error('Geolocation is not supported by this browser.');

    }

}

getLocation();

navigator.geolocation.watchPosition(

    (position) => {

        const locationInfo = document.getElementById('locationInfo');

        const latitude = position.coords.latitude;

        const longitude = position.coords.longitude;

        locationInfo.textContent = `Location: ${latitude}, ${longitude}`;

    },

    (error) => {

        console.error('Error watching location:', error.message);

    }

);

CSS file:

#gallery {

    display: flex;

    flex-wrap: wrap;

}

img {

    width: 150px;

    height: 150px;

    object-fit: cover;

    border: 1px solid #ddd;

}

.location-info {

    position: fixed;

    top: 10px;

    right: 10px;

    background-color: #fff;

    padding: 10px;

    border: 1px solid #ddd;

}

**Виконання програми:**

****

GitHub:

https://github.com/djproblematic/frontend