

Département Mathématiques et Informatique

« Génie du Logiciel et des Systèmes Informatiques Distribués »

2^{ème} année

**Rapport de Projet Développement
Multimédia**

Réalisé par :

- AIT LHADJ Nouhaila.

Année universitaire : 2018-2019

I. Projet développement multimédia :

1. Enoncé :

Nous nous intéressons au développement d'un module permettant de capturer la photo de l'utilisateur pour pouvoir l'ajouter à son profil. Ce module sera utilisable dans les applications Web, Mobiles hybrides et Desktop basées sur les technologies WEB.

Le module sera développé en utilisant une approche incrémentale en trois versions successives :

Version 1 :

Cette version doit comporter :

1. Etape 1 : Une interface pour les actions de la capture
2. Etape 2 : Enregistrement de l'image encodée

Version 2 :

Cette version va améliorer l'interface de la version 1 en ajoutant :

1. Etape 1 : Ajouter à l'interface des composants permettant d'ajuster la qualité de l'image en ajoutant des effets à l'image (contraste, luminosité, teinte, etc...)

Version 3 :

Dans cette version, nous permettrons aux utilisateurs de recadrer et de pivoter l'image (cropping). Pour cela, l'interface doit présenter un rectangle (Outline) avec des poignets permettant de guider l'utilisateur dans ses actions.

1. Etape 1 : Tracer l'Outline avec ses poignets de recadrage
2. Etape 2 : Rendre l'Outline et ses poignets sensibles aux mouvements de la souris (Evénements)
3. Etape 3 : Gérer le recadrage (en largeur et en hauteur) et le pivotement selon les événements de la souris
4. Etape 4 : Enregistrement de l'image recadrer.

2. Version 1 :

Dans cette version l'utilisateur appuie sur le bouton « photo » dans l'accueil pour afficher l'interface qui contient une « vidéo » et le bouton « capturer ». La vidéo affiche la sortie de la caméra de l'utilisateur.

L'interface d'Accueil, qui contient le bouton **Photo** :



Dans cette interface nous avons utilisé dans code html :

```
<div id="interface1">
<button id="btn1" class="camera" onclick="displayInterface2()" >Photo</button>
</div>
```

L'événement **OnClick** qui contient la fonction « **displayInterface2()** », dans le fichier JavaScript :

```

//apres avoir cliquer sur photo
function displayInterface2(){
    document.getElementById('centerdiv').style.display = 'block';
    document.getElementById('btn1').style.display = 'none';
    video = document.getElementById('camera');
    if (navigator.mediaDevices && navigator.mediaDevices.getUserMedia) {
        navigator.mediaDevices.getUserMedia({ video: true }).then(function (stream) {
            video.srcObject = stream;
            video.play();
            //fonctio pour recuperer la video
        }, function () {
            let source = document.createElement('source');
            source.type = 'video/mp4';
            source.src = 'video/loading.mp4';
            video.appendChild(source);
            video.load();
            video.currentTime = 0;
            video.loop = true;
            video.play();
        });
    }
}

```

Si le dispositif de l'utilisateur n'a pas de caméra, on affiche et on démarre une petite vidéo en boucle.

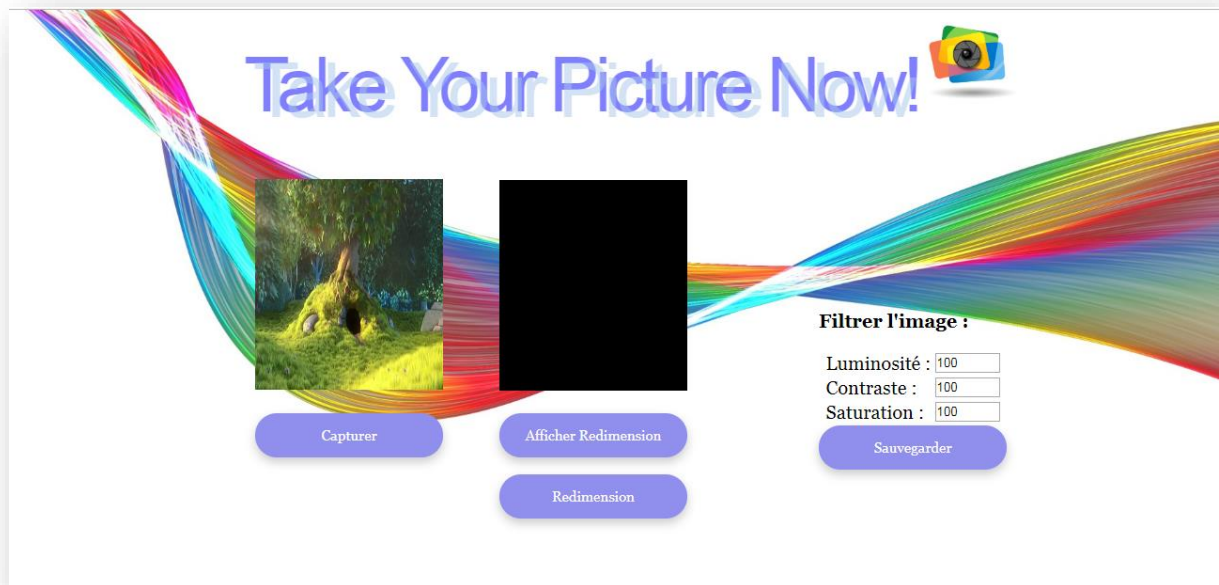
Et ça montre dans le code :

```

function () {
    let source = document.createElement('source');
    source.type = 'video/mp4';
    source.src = 'video/loading.mp4';
    video.appendChild(source);
    video.load();
    video.currentTime = 0;
    video.loop = true;
    video.play();
}

```

La vidéo « **Loading.mp4** » va être affichée si la caméra n'existe pas.



Quand l'utilisateur appuie sur le bouton « capturer », on affiche un Count down de 3 secondes au terme duquel on capture l'image. (On l'affiche).

Dans le code html, le bouton Capturer contient un événement **OnClick** qui est basé sur une fonction Java Script « **Capture()** » :

```
<div id="opvid">
<h3></h3>
<button class="camera2" onclick='capture()'>Capturer</button>

</div>
```

Dans Java Script :

```

async function capture() {
  //affichage de compteur de 3->0 avant de capturer l image
  showCountdown(true);
  ms=3;
  do {
    document.getElementById('countdown').innerText = "" + ms;
    await sleep(1000);
    ms--;
  }while(ms >= 0);
  showCountdown(false);
  canvas = document.getElementById('resultat');
  //notre CANVAS
  context = canvas.getContext('2d');
  video = document.getElementById('camera');

  context.filter = "none";
  //recuperer picture dans canvas
  context.drawImage(video, 0, 0, canvas.clientWidth, canvas.clientHeight);
  image = new Image();
  image.src = canvas.toDataURL();
  resize();
}

```

Cette fonction elle-même utilise la une autre fonction pour afficher le Countdown, « **showCountdown** ».

```

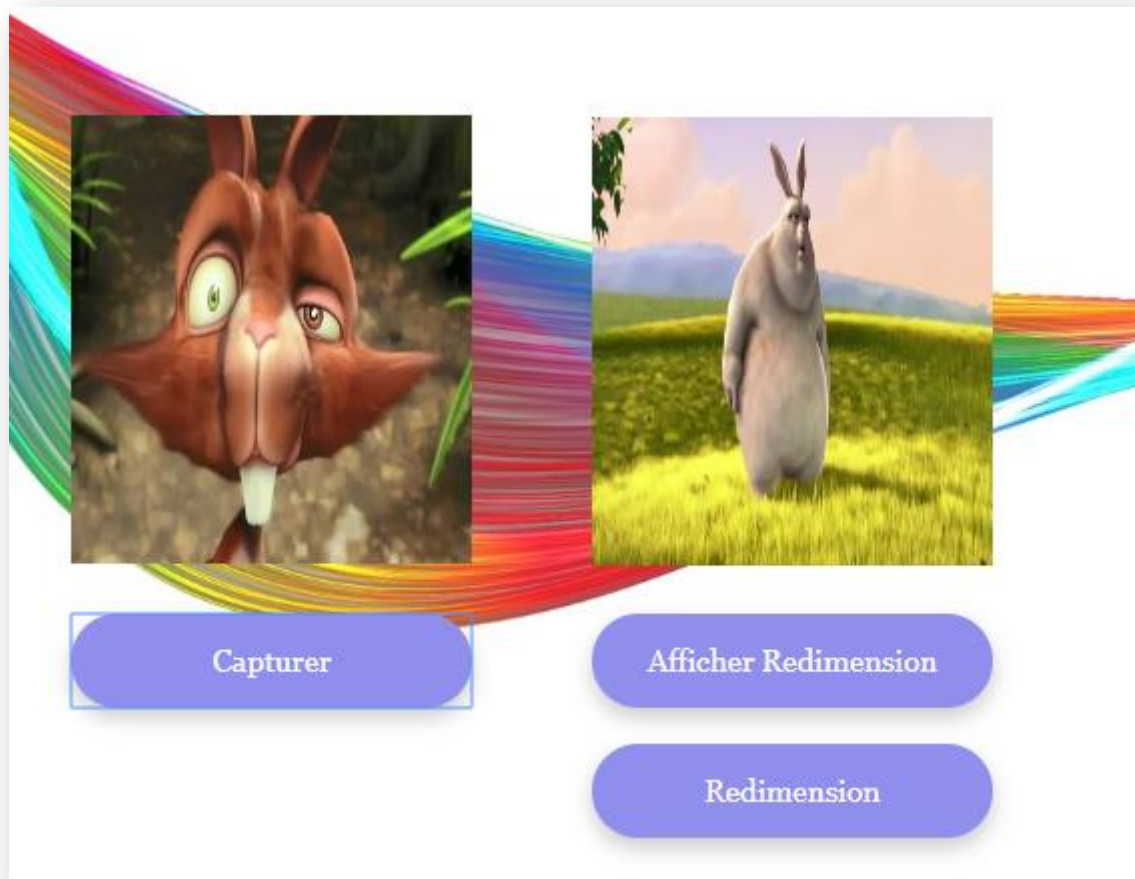
//display countdown
function showCountdown(bool) {
  if(bool) {
    document.getElementById('countdown').style.display = 'block';
  }else{
    //apres 0 disparaitre le countdown
    document.getElementById('countdown').style.display = 'none';
  }
}

```

On clique sur Capturer le **CountDown** s'affiche :



Après c'est récupérée le **Canvas** :






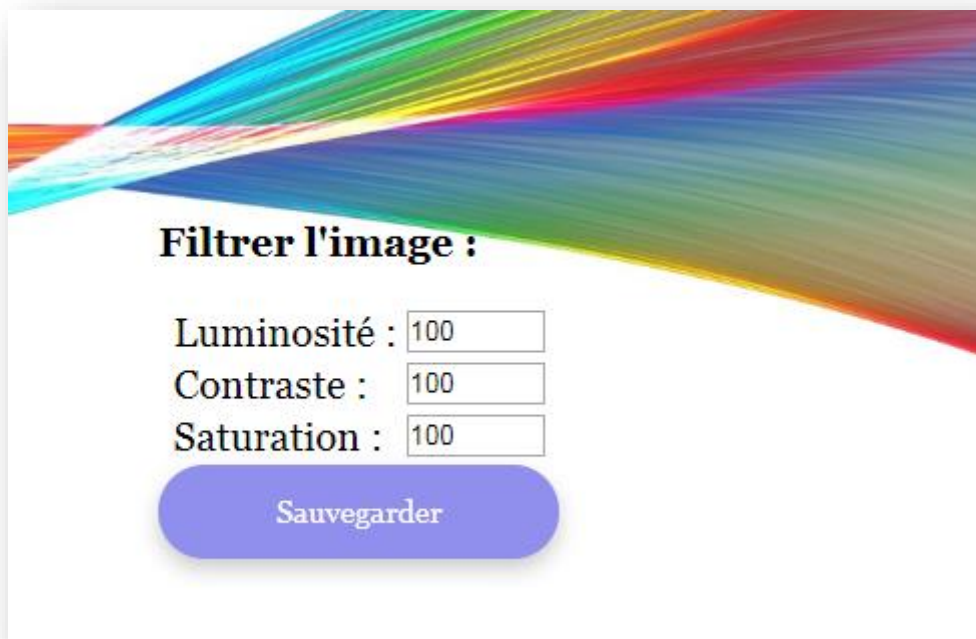
L'image est ensuite encodée sous format Base64, pour être téléchargé après.

3. Version 2 :

Dans cette version on va améliorer l'interface de la version 1 en ajoutant des composants permettant d'ajuster la qualité de l'image en ajoutant des effets à l'image (contraste, luminosité, teinte, etc....).

Mon projet contient 3 paramètres :

-  Contraste.
-  Luminosité.
-  Teinte.



- Le premier paramètre c'est la luminosité qui prend une valeur de 0 à 200.
- Le 2 ème paramètre c'est la contraste qui prend une valeur de 0 à 200.
- Le 3 ème paramètre c'est la saturation qui prend une valeur de 0 à 200.

La fonction qui permet de faire ça en Java Script :

```
function filtercanvas(id, val) {
    //les filtres varient entre 0 et 200
    //luminosité
    let val1 = parseFloat(document.getElementById('fbrightness').value);
    //contraste
    let val2 = parseFloat(document.getElementById('fcontrast').value);
    //saturation
    let val3 = parseFloat(document.getElementById('fsaturate').value);
    switch (id) {
        case "fbrightness": {
            val1 = val;
            break;
        }
        case "fcontrast": {
            val2 = val;
            break;
        }
        case "fsaturate": {
            val3 = val;
            break;
        }
    }
    context.filter = "brightness(" + val1 + "%) contrast(" + val2 + "%) saturate(" + val3 + "%)";
    context.drawImage(image, 0, 0, canvas.offsetWidth, canvas.offsetHeight);
}

```

✚ Exemple d'application sur un exemple:

Avant modification :



Après modification :



4. Version 3 :

Dans cette version, nous permettrons aux utilisateurs de recadrer l'image.

Pour cela, l'interface doit présenter un rectangle (Outline) avec des poignets permettant de guider l'utilisateur dans ses actions.

Si on clique sur le bouton « **afficher redimension** », un carreau noir avec 8 points s'affiche, permet nous de redimensionner l'image comme on veut.

Dans le code html le bouton contient un événement **OnClick** avec la fonction dans JS

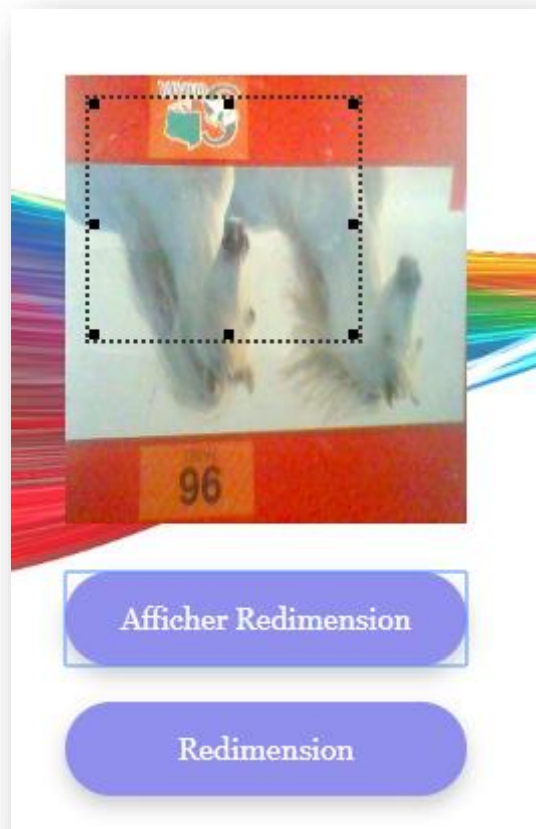
showCarre(true) :

```
<button class="camera2" onclick="showCarre(true)">Afficher Redimension</button><br /><br />
```

Dans le code Java Script :

```
//affichage de carree de redimension
function showCarre(bool) {
    if(bool) {
        document.getElementById('cadre').style.display = 'block';
    }else{
        document.getElementById('cadre').style.display = 'none';
    }
}
```

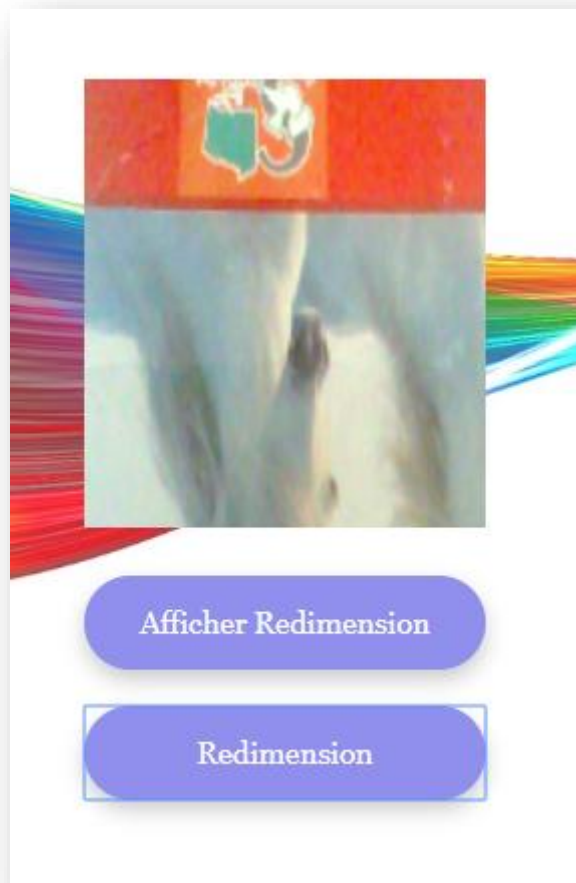
Et s'affiche dans l'image :



Après si on clique sur **Redimension**, pour redimensionner, le bouton contient la fonction de redimensionnement :

```
function recadrage() {
    //apparition de la nouvelle picture avec redimension
    let elm = document.getElementById('cadre');
    context.drawImage(image, valueInCss(elm, 'left'), valueInCss(elm, 'top'), valueInCss(elm, 'width'), valueInCss(elm, 'height'));
    image = new Image();
    image.src = canvas.toDataURL();
    //on fait disparaître le carrée de dimensionnement
    showCarre(false);
}
```

Le résultat si on clique sur **Redimension** :



Finalement si on clique sur le bouton **Sauvegarder**, l'image va être enregistrée, avec la fonction :

```
//pour le téléchargement de l'image en base 64
function downloadImageBase64() {
    const imgsrc = canvas.toDataURL();
    download(imgsrc);
}
```

Qui fait appel à la fonction :

```
//il creer une pic qu'on voit pas il la download et l'enregistre  
function download(url) {  
    let element = document.createElement('a');  
    element.setAttribute('href', url);  
    element.setAttribute('download', 'image.png');  
    element.style.display = 'none';  
    document.body.appendChild(element);  
    element.click();  
    document.body.removeChild(element);  
}
```

Et il commence à se télécharger dans le bas de la page :

