Documentation de Stage de 1ère Année

Projet de Compression d'Images pour Boutique en Ligne

Yassine Tamani

17 décembre 2024

Table des matières

1 Introduction **2**

2 Objectifs **3**

3 Technologies Utilisées **4**

3.1 Front-end 4

3.2 Back-end 4

4 Architecture du Projet **5**

5 Exemple d'Interface **6**

6 Extrait de Code côté Serveur (PHP) **7**

7 Extrait de Code côté Client (JavaScript) **10**

8 Dépôt GitHub **11**

9 Conclusion **12**

Chapitre 1

Introduction

Dans le cadre de mon stage de première année, j'ai développé une solution de compression d'images afin d'améliorer les performances de chargement des pages d'une boutique en ligne. Réduire la taille des images (principalement JPEG) permet d'accélérer le temps de chargement, offrant une meilleure expérience utilisateur, surtout sur mobile, et améliorant potentiellement le référencement (SEO) et le taux de conversion du site.

Chapitre 2

Objectifs

L'objectif principal était de :

- Automatiser la compression des images (JPEG) sans perte de qualité notable.
- Améliorer la vitesse de chargement des pages.
- Offrir une interface utilisateur simple, permettant de traiter plusieurs images simultanément.

Chapitre 3

Technologies Utilisées

3.1 Front-end

— **HTML5** : Structure sémantique des pages.
— **Tailwind CSS** : Mise en page responsive.
— **JavaScript** : Gestion des interactions, drag & drop, téléchargement ZIP.

3.2 Back-end

— **PHP** : Traitement et compression des images via GD.

Chapitre 4

Architecture du Projet

— **Interface de chargement :** Import jusqu'à 20 images JPEG.
— **Script PHP (`compress.php`) :** Compression, redimensionnement, conversion en base64.
— **UI JS :** Affichage du ratio de compression, téléchargement individuel ou groupé (ZIP).

Chapitre 5

Exemple d'Interface

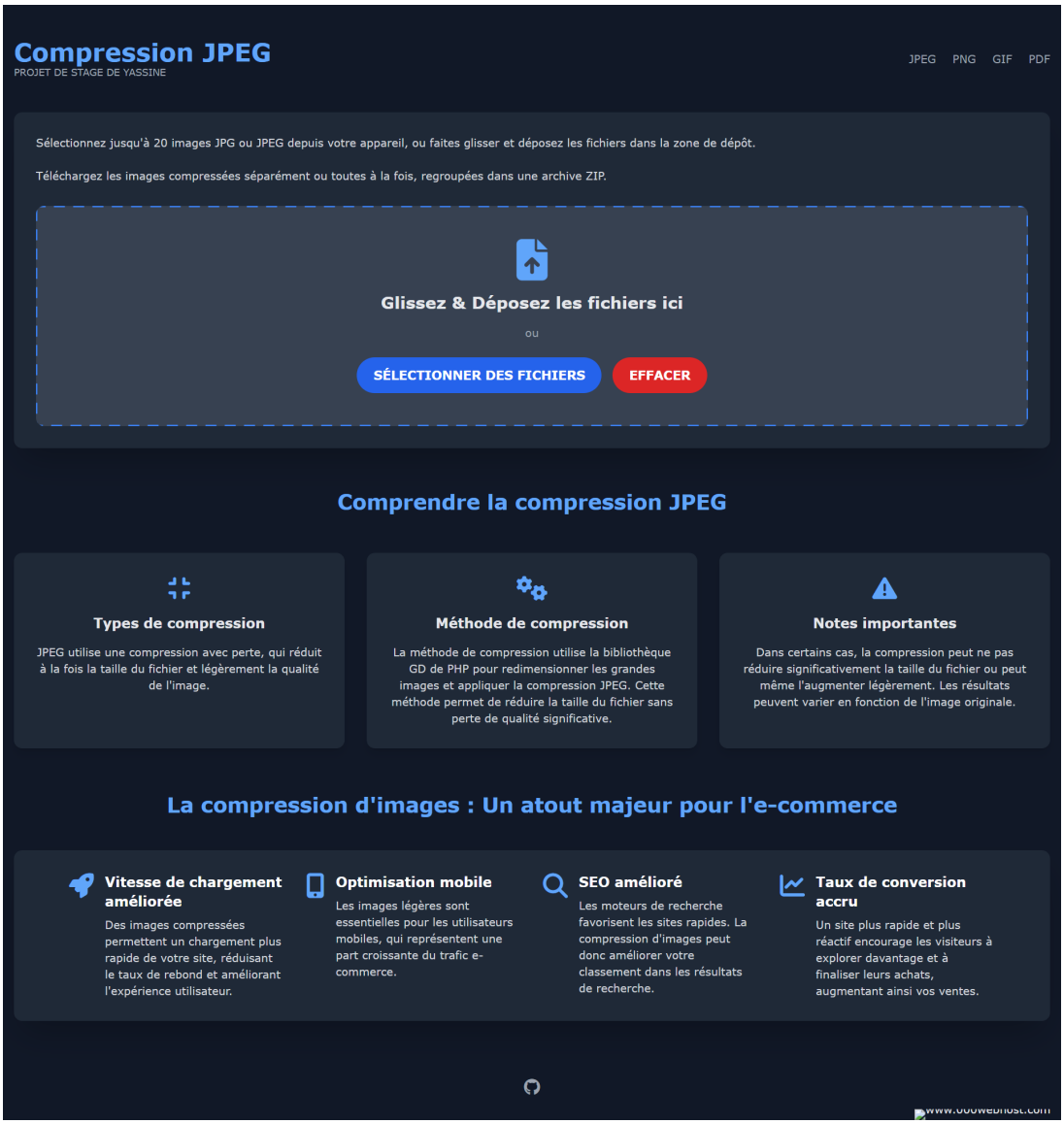

FIGURE 5.1 – Aperçu de l'interface de compression d'images.

Chapitre 6

Extrait de Code côté Serveur (PHP)

```php
<?php
ini_set('memory_limit', '256M');
header('Content-Type: application/json');

if ($_SERVER['REQUEST_METHOD'] !== 'POST' || !isset($_FILES['image'])) {
    echo json_encode(['error' => 'Invalid request']);
    exit;
}

$file = $_FILES['image'];
$originalSize = $file['size'];

if ($file['error'] !== UPLOAD_ERR_OK) {
    echo json_encode(['error' => 'File upload failed']);
    exit;
}

$info = getimagesize($file['tmp_name']);
if ($info === false) {
    echo json_encode(['error' => 'Invalid image file']);
    exit;
}

$type = $info[2];
$quality = 85;

switch ($type) {
    case IMAGETYPE_JPEG:
        $image = imagecreatefromjpeg($file['tmp_name']);
        break;
    case IMAGETYPE_PNG:
        $image = imagecreatefrompng($file['tmp_name']);
        imagealphablending($image, true);
        imagesavealpha($image, true);
        break;
    case IMAGETYPE_GIF:
        $image = imagecreatefromgif($file['tmp_name']);
        break;
```

```
    default:
        echo json_encode(['error' => 'Unsupported image type']);
        exit;
}

$maxWidth = 1920;
$maxHeight = 1080;
$width = imagesx($image);
$height = imagesy($image);

if ($width > $maxWidth || $height > $maxHeight) {
    $ratio = min($maxWidth / $width, $maxHeight / $height);
    $newWidth = round($width * $ratio);
    $newHeight = round($height * $ratio);
    $newImage = imagecreatetruecolor($newWidth, $newHeight);

    if ($type == IMAGETYPE_PNG) {
        imagealphablending($newImage, false);
        imagesavealpha($newImage, true);
        $transparent = imagecolorallocatealpha($newImage, 255, 255, 255,
   127);
        imagefilledrectangle($newImage, 0, 0, $newWidth, $newHeight,
   $transparent);
    }

    imagecopyresampled($newImage, $image, 0, 0, 0, 0, $newWidth, $newHeight
   , $width, $height);
    $image = $newImage;
}

ob_start();
imagejpeg($image, null, $quality);
$imageData = ob_get_clean();
imagedestroy($image);

$compressedSize = strlen($imageData);
$base64Image = base64_encode($imageData);

echo json_encode([
    'base64Image' => '' . $base64Image,
    'originalSize' => $originalSize,
    'compressedSize' => $compressedSize,
    'compressionRatio' => round(($originalSize - $compressedSize) /
   $originalSize * 100, 2) . '%'
```

```
79 ]);
```

Listing 6.1 – Extrait du code PHP (`compress.php`)

Chapitre 7

Extrait de Code côté Client (JavaScript)

```
document.addEventListener('DOMContentLoaded', function() {
    const fileInput = document.getElementById('fileInput');
    const clearButton = document.getElementById('clearButton');
    const downloadAllButton = document.getElementById('downloadAllButton');
    const dropZone = document.getElementById('dropZone');
    const resultContainer = document.getElementById('resultContainer');
    // Gestion du drag & drop, compression, affichage des r sultats...
});
```

Listing 7.1 – Extrait JS pour l'UI

Chapitre 8

Dépôt GitHub

Le code source complet du projet est disponible ici :

`https://github.com/rabbyt3s/CompressJPEG`

Chapitre 9

Conclusion

Ce projet permet de compresser efficacement les images, réduisant le temps de chargement du site et améliorant l'expérience utilisateur, le SEO et les conversions. L'interface, simple et intuitive, facilite l'usage et l'implémentation de cette solution dans un contexte e-commerce.