**App.py**

**Overview**

This code is a Flask web application that allows users to upload, compress, and download image and video files. It also provides functionality to delete files. The application handles both image and video compression, with specific handling for different file types.

**Libraries and Imports**

* import os: Imports the OS module for interacting with the operating system.
* from flask import Flask, request, redirect, url\_for, render\_template, send\_from\_directory, send\_file: Imports various functions from Flask for creating the web application.
* from werkzeug.utils import secure\_filename: Imports a function to secure filenames.
* from PIL import Image: Imports the Python Imaging Library (PIL) for image processing.

**Flask Application Setup**

* app = Flask(\_\_name\_\_): Initializes a Flask application.
* UPLOAD\_FOLDER = 'uploads': Defines the folder to store uploaded files.
* COMPRESSED\_FOLDER = 'compressed': Defines the folder to store compressed files.
* ALLOWED\_EXTENSIONS = {'png', 'jpg', 'jpeg', 'mp4', 'avi', 'mov'}: Specifies allowed file extensions.
* app.config['UPLOAD\_FOLDER'] = UPLOAD\_FOLDER: Sets the upload folder in the Flask app configuration.
* app.config['COMPRESSED\_FOLDER'] = COMPRESSED\_FOLDER: Sets the compressed folder in the Flask app configuration.
* os.makedirs(UPLOAD\_FOLDER, exist\_ok=True): Ensures the upload folder exists.
* os.makedirs(COMPRESSED\_FOLDER, exist\_ok=True): Ensures the compressed folder exists.

**Utility Functions**

* def allowed\_file(filename): Checks if a file is allowed based on its extension.
  + Returns True if the file extension is in the allowed list, otherwise False.
* def compress\_image(image\_path, output\_path, quality=60): Compresses an image and saves it with the specified quality.
  + Opens the image using PIL.
  + Converts RGBA images to RGB if necessary.
  + Saves the image to the specified output path with the given quality.

**Routes**

* @app.route('/'): Renders the home page.
  + def index(): Returns the index.html template.
* @app.route('/upload', methods=['POST']): Handles file upload.
  + Checks if a file is included in the request.
  + Checks if the filename is valid and allowed.
  + Saves the file to the upload folder.
  + Returns the index.html template with upload confirmation and file size.
* @app.route('/compress', methods=['POST']): Compresses the uploaded file.
  + Determines the file path and compressed file path.
  + Compresses images using compress\_image.
  + Compresses videos using ffmpeg (requires ffmpeg to be installed).
  + Returns the index.html template with compression confirmation and file sizes.
* @app.route('/uploads/<filename>'): Serves the uploaded file.
  + def uploaded\_file(filename): Sends the uploaded file from the upload folder.
* @app.route('/compressed/<filename>'): Serves the compressed file.
  + def compressed\_file(filename): Sends the compressed file from the compressed folder.
* @app.route('/download/<filename>'): Provides a download link for the compressed file.
  + def download\_file(filename): Sends the compressed file as an attachment for download.
* @app.route('/delete/<filename>', methods=['POST']): Deletes the compressed file.
  + Deletes the compressed file if it exists.
  + Deletes the uploaded file if it exists.
  + Redirects to the home page.

**Running the Application**

* if \_\_name\_\_ == '\_\_main\_\_': Ensures the app runs only if the script is executed directly.
  + app.run(debug=True): Runs the Flask app in debug mode.

Templates/index.html

**Overview**

This HTML template is for a Flask web application that allows users to upload images and videos, compress them, and download the compressed files. The template uses Bootstrap for styling and ensures responsiveness across devices.

**HTML Structure**

* <!DOCTYPE html>: Declares the document type and version of HTML.
* <html lang="en">: Starts the HTML document and sets the language to English.

**Head Section**

* <head>: Contains meta-information about the document.
  + <meta charset="UTF-8">: Sets the character encoding to UTF-8.
  + <meta name="viewport" content="width=device-width, initial-scale=1.0">: Ensures the webpage is responsive.
  + <title>Image and Video Compression</title>: Sets the title of the webpage.
  + <link rel="stylesheet" href="{{ url\_for('static', filename='styles.css') }}">: Links to a custom CSS file for additional styling.
  + <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">: Links to Bootstrap's CSS for styling.

**Body Section**

* <body>: Contains the main content of the webpage.
  + <div class="container">: A Bootstrap container for centering the content and adding padding.
    - <h1>Image and Video Compression</h1>: A header for the webpage.
    - <form action="/upload" method="post" enctype="multipart/form-data">: A form for uploading files.
      * <input type="file" name="file" class="form-control mb-2">: An input field for file selection with Bootstrap styling.
      * <button type="submit" class="btn btn-primary">Upload</button>: A submit button for the form.

**Conditional Content Display**

* {% if filename %}: Checks if a file has been uploaded.
  + <div class="content mt-4">: A container for the content with a top margin.
    - <div class="row">: A Bootstrap row to align columns.
      * <div class="col-md-6">: A Bootstrap column for the original content.
        + <h2>Original Content (Size: {{ file\_size }} KB)</h2>: Displays the size of the uploaded file.
        + {% if filename.lower().endswith(('.png', '.jpg', '.jpeg')) %}: Checks if the file is an image.

<img src="{{ url\_for('uploaded\_file', filename=filename) }}" alt="Uploaded Image" class="img-fluid">: Displays the uploaded image.

* + - * + {% elif filename.lower().endswith(('.mp4', '.avi', '.mov')) %}: Checks if the file is a video.

<video controls class="img-fluid">: Displays the uploaded video with controls.

<source src="{{ url\_for('uploaded\_file', filename=filename) }}" type="video/mp4">: Specifies the video source.

Your browser does not support the video tag.: Fallback message for unsupported browsers.

* + - * + <form action="/compress" method="post" class="mt-2">: A form to trigger file compression.

<input type="hidden" name="filename" value="{{ filename }}">: Hidden input to pass the filename.

<button type="submit" class="btn btn-warning">Compress</button>: A button to submit the compression form.

* {% if compressed %}: Checks if the file has been compressed.
  + <div class="col-md-6">: A Bootstrap column for the compressed content.
    - <h2>Compressed Content (Size: {{ compressed\_size }} KB)</h2>: Displays the size of the compressed file.
    - {% if compressed\_filename.lower().endswith(('.png', '.jpg', '.jpeg')) %}: Checks if the compressed file is an image.
      * <img src="{{ url\_for('compressed\_file', filename=compressed\_filename) }}" alt="Compressed Image" class="img-fluid">: Displays the compressed image.
    - {% elif compressed\_filename.lower().endswith(('.mp4', '.avi', '.mov')) %}: Checks if the compressed file is a video.
      * <video controls class="img-fluid">: Displays the compressed video with controls.
        + <source src="{{ url\_for('compressed\_file', filename=compressed\_filename) }}" type="video/mp4">: Specifies the video source.
        + Your browser does not support the video tag.: Fallback message for unsupported browsers.
    - <a href="{{ url\_for('compressed\_file', filename=compressed\_filename) }}" class="btn btn-success mt-2" download>Download</a>: A download link for the compressed file.
    - <form action="/delete/{{ compressed\_filename }}" method="post" class="d-inline-block mt-2">: A form to delete the compressed file.
      * <button type="submit" class="btn btn-danger">Delete</button>: A button to submit the delete form