Online Image Compressor

Computer Networks Project

Team members

1. Adithya Suresh - AM.EN.U4CSE18102
2. Adithyan S Raj - AM.EN.U4CSE17510

Introduction

This project is aimed at users who want to compress their JPEG files (file size < 50Mb) to a specific size. After compression, we allow the users to either download the file from our server or get the files sent to their email address. This is a user-friendly web application in Python.

We use FTP (File Transfer Protocol) for upload and download of the file and SMTP (Simple Mail Transfer Protocol) for sending the file via email.

Modules

1. **File Upload**: The user can upload a .jpeg file and it will be sent to the server for compression.
2. **File Download**: Allows the user to download the file after compression.
3. **Server**: Compresses the files it receives and sends it back to the client. Optionally if the uploader chooses to send the file to an email address, the server will ask for the email address first.

User Guide

1. Select a **.jpg** or **.jpeg** image to upload.
2. Click on the **upload** button.
3. The file will not get uploaded in the following cases:
    a. if the file extension is not .jpg or .jpeg
    b. If the file size **exceeds 50MB**
4. If file gets accepted, you will get redirected to a new page.
5. On successful compression (to < **80% of original size**), the user has two options:
    a. The download button downloads the compressed image for the user.
    b. Send as email button sends the compressed image as an email attachment to a valid and existing email address specified by the user.

Development

Frontend

All pages are made using HTML/CSS/JavaScript and Jinja (a templating engine)

1. **Index Page**: Functions as a User Guide
2. **Image upload**: choose file to upload and upload for compression
3. **Compression status**: This page tells the user whether the compression was successful or not. On successful compression, it shows the compressed image, a download button and a send file as email button.
4. **Email sent**: A simple page notifying the user that the email has been sent successfully.

Backend

1. **Flask**: a micro web framework written in Python.
2. **Flask-mail**: an extension (package) to the Flask framework for sending emails.
3. **smtp.gmail.com**: a free SMTP server (port 465).
4. **Heroku**: a cloud platform to deploy web apps.

Deployment

Deployment was done on Heroku.

1. Create a file called wsgi.py for starting the app on the cloud.
2. Start a terminal and run Heroku CLI.
3. Set/enter the login credentials
4. Run the command

   ```
   heroku create <app-name>
   ```

   after setting `<app-name>` to a preferred name, say online-jpeg-compressor (without the angular braces).

5. The app will get deployed on http://online-jpeg-compressor.herokuapp.com

Software

Language Stack: Python, HTML, CSS, JavaScript

Frameworks: Flask(backend), OpenCV, Numpy

Text Editor: Atom, VS Code

Cloud Hosting: Heroku

Version Control: Git and Github

Link to our Project

1. Source Code (Github)
2. Our app

References

1. Uploading Files with Flask

Thank you