Design Document

Iterator Design Pattern - Can be used to perform a set of instructions sequentially based on a list. While relating to Individual Project, as the client should apply for single API call which should perform a list of actions as already selected by client. This design pattern seems like a viable option for Individual Project implementation.

Chain of Responsibility – Can be used to perform a set of instructions based on who will perform the next task. If multiple functions are created for each type of task like one function Rotate Image to Left and another to rotate right. This will pass the updated image to the next function in line and the last function will return the latest modified image to the client through the API

Prototype – Can be used to perform thumbnail creation tasks, where is the original file is cloned and further actions are performed without updating the original file. So, when the thumbnails are supposed to be created, the original file will not be updated but a clone of that image file will be created, and thumbnail is created from that cloned file and then sent to the client through the API.

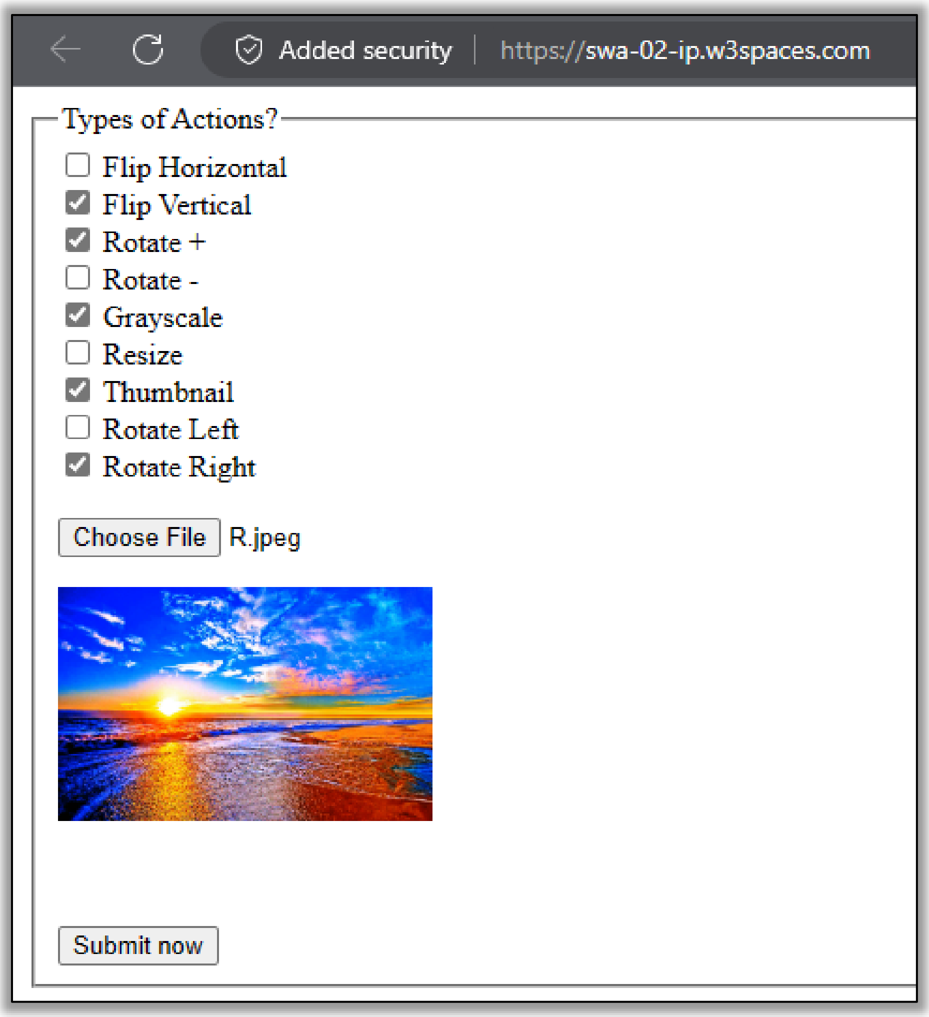
Façade – Can be used to perform multiple small actions through subclasses and the main class remains the façade of the tasks. So, when list of actions is provided to the API from the client, API will be the façade and the various small functions within the API will perform small tasks like rotate left, rotate right etc.

All the above-mentioned design patterns are good in their own way and have their own pros and cons to implement in this individual project.
However, I would like to use a combination of design patterns to implement the Individual project, i.e., Façade and Prototype. Wherein Façade will be used for API as main façade and various functional tasks as subsystems and specifically for thumbnail task prototype design will be used .

Little Language

Form is designed with HTML along with image upload button

- On submit all data is fetch in JavaScript
- Image is set inside the Form Data
- XML Request object
- Post request opens new URL with parameters



Types of Actions?

☐ Flip Horizontal
☑ Flip Vertical
☑ Rotate +
☐ Rotate -
☑ Grayscale
☐ Resize
☑ Thumbnail
☐ Rotate Left
☑ Rotate Right

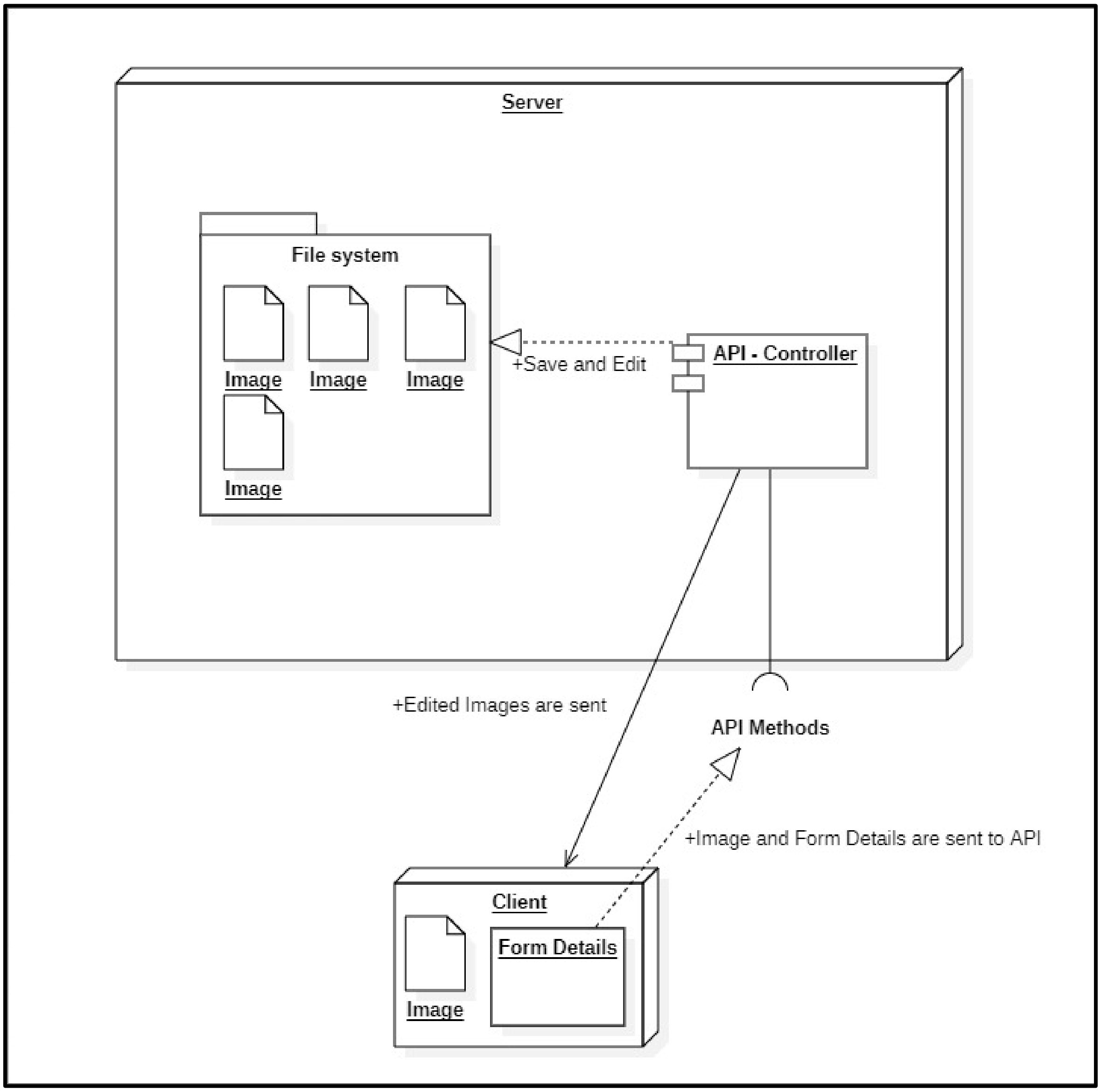
R.jpeg

- Once the request objects is set, the request is sent to the server and wait for response.

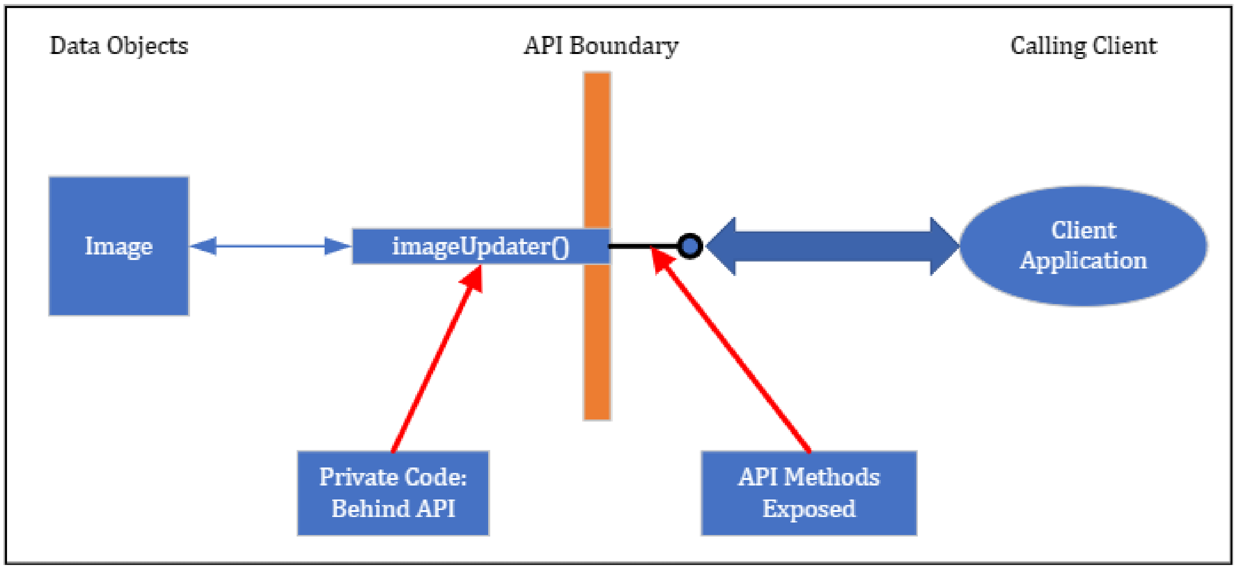
```
{horizontal: false, vertical: true, image: FileList, grayscale: true, resize: false, …}
        grayscale: true
        horizontal: false
        image: FileList
                0: File {name: '2203efe3-8e84-4aeb-b30c-
                e57c38ddac04.jpg_share.jpg', lastModified: 1615287659000, lastModifiedDate: Tue
                Mar 09 2021 03:00:59 GMT-0800 (Pacific Standard
                Time), webkitRelativePath: '', size: 114470, …}
                length: 1
                [[Prototype]]: FileList
        resize: false
        rotateDecr: true
        rotateIncr: true
        rotateLeft: true
        rotateRight: true
        thumbnail: false
        vertical: true
```

- As soon as the response is arrived new image is displayed next to original image in HTML

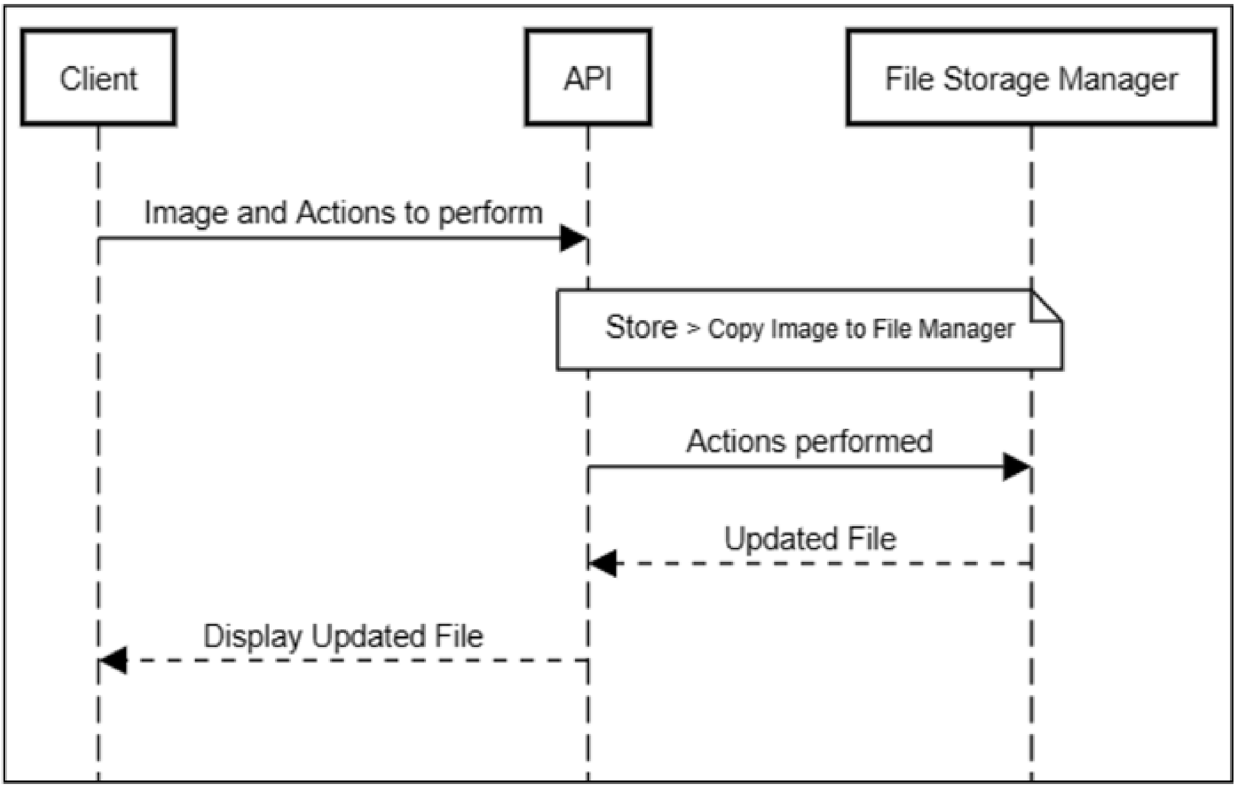
System Diagram :-



Context Diagram



Sequence Diagram



Server-Side Code / File Name :- IP_Flask_Server.py

Run from Terminal using command –

```
        Flask --app IP_Flask_Server.py run
```

```
from PIL import Image, ImageOps
import re
from flasgger import Swagger, LazyString, LazyJSONEncoder, swag_from
from flask import Flask, flash, render_template, request, redirect, url_for, jsonify, send_file
import json
import os
from werkzeug.utils import secure_filename
from werkzeug.datastructures import  FileStorage

app = Flask(__name__)
app.config["DEBUG"] = True
app.config['UPLOAD_FOLDER'] = 'static/uploads/'

@app.route('/ipMain', methods=['GET'])
def ipMain():
    return render_template('ipMain.html')

@app.route('/editImage',methods = ['GET', 'POST'])
def editImage():  # sourcery skip: avoid-builtin-shadow
    actions = request.form.getlist('ListOfActions')
    resizeHeight = int(request.form.getlist('Height')[0])
    resizeWidth = int(request.form.getlist('Width')[0])
    posted_file = request.files.get('imageFile', '')
    posted_file_name = request.files.get('imageFile', '').filename
    # x = re.split(", |_|-|!|\+",actions[0])
    x = list(actions[0].replace(",","").replace(" ",""))
    print('============= List Of Actions =============', actions)
    print('============= Resize Height =============', resizeHeight)
    print('============= Resize Width =============', resizeWidth)
    print("============= X =============",x)
    print('============= FileStorage =============', posted_file)
    print('============= Name =============', posted_file_name)

    file = FileStorage(posted_file)
    file.save(os.path.join(app.config['UPLOAD_FOLDER'],posted_file_name))
    pil_image = Image.open(file)

    for i in range(len(x)):
        if x[i] == "1":
            #1. Flip Horizontal
            pil_image = ImageOps.mirror(pil_image)
            pil_image.save(f'static/uploads/Seq_{str(i)}_Task_{str(x[i])}_{posted_file_name}',quality = 95)
        if x[i] == "2":
            #2. Flip Vertical
            pil_image = ImageOps.flip(pil_image)
            pil_image.save(f'static/uploads/Seq_{str(i)}_Task_{str(x[i])}_{posted_file_name}',quality = 95)
        if x[i] == "3":
            #3. Rotate +45 Degrees(Clock Wise)
            pil_image = pil_image.rotate(45, resample=Image.BICUBIC)
            pil_image.save(f'static/uploads/Seq_{str(i)}_Task_{str(x[i])}_{posted_file_name}',quality = 95)
        if x[i] == "4":
            #4. Rotate -45 Degrees(Anti-Clock Wise)
            pil_image = pil_image.rotate(-45, resample=Image.BICUBIC)
            pil_image.save(f'static/uploads/Seq_{str(i)}_Task_{str(x[i])}_{posted_file_name}',quality = 95)
        if x[i] == "5":
            #5. Convert to Grayscale
            pil_image = ImageOps.grayscale(pil_image)
            pil_image.save(f'static/uploads/Seq_{str(i)}_Task_{str(x[i])}_{posted_file_name}',quality = 95)
        if x[i] == "6":
            #6. Resize Image
            pil_image = ImageOps.exif_transpose(pil_image.resize((resizeHeight, resizeWidth), Image.Resampling.LANCZOS))
            pil_image.save(f'static/uploads/Seq_{str(i)}_Task_{str(x[i])}_{posted_file_name}',quality = 95)
        if x[i] == "7":
            #7. Thumbnail(300px X 300px)
            thumbnail_image = ImageOps.exif_transpose(pil_image.resize((300, 300), Image.Resampling.LANCZOS))
            thumbnail_image.save(f'static/uploads/Seq_{str(i)}_Task_{str(x[i])}_{posted_file_name}',quality = 95)
        if x[i] == "8":
            #8. Rotate Right(90 Degrees - Clock Wise)
            pil_image = pil_image.rotate(90, resample=Image.BICUBIC)
            pil_image.save(f'static/uploads/Seq_{str(i)}_Task_{str(x[i])}_{posted_file_name}',quality = 95)
        if x[i] == "9":
            #9. Rotate Left(90 Degrees - Anti-Clock Wise)
            pil_image = pil_image.rotate(-90, resample=Image.BICUBIC)
            pil_image.save(f'static/uploads/Seq_{str(i)}_Task_{str(x[i])}_{posted_file_name}',quality = 95)
        else:
            print("in Else")

    imageListPath = os.listdir('static/uploads')
    imageList = [f'uploads/{image}' for image in imageListPath]
    print('============= Name =============', imageList)
    return render_template("temp.html", imageList=imageList)
```

Client-Side Code 1 - ipMain.html

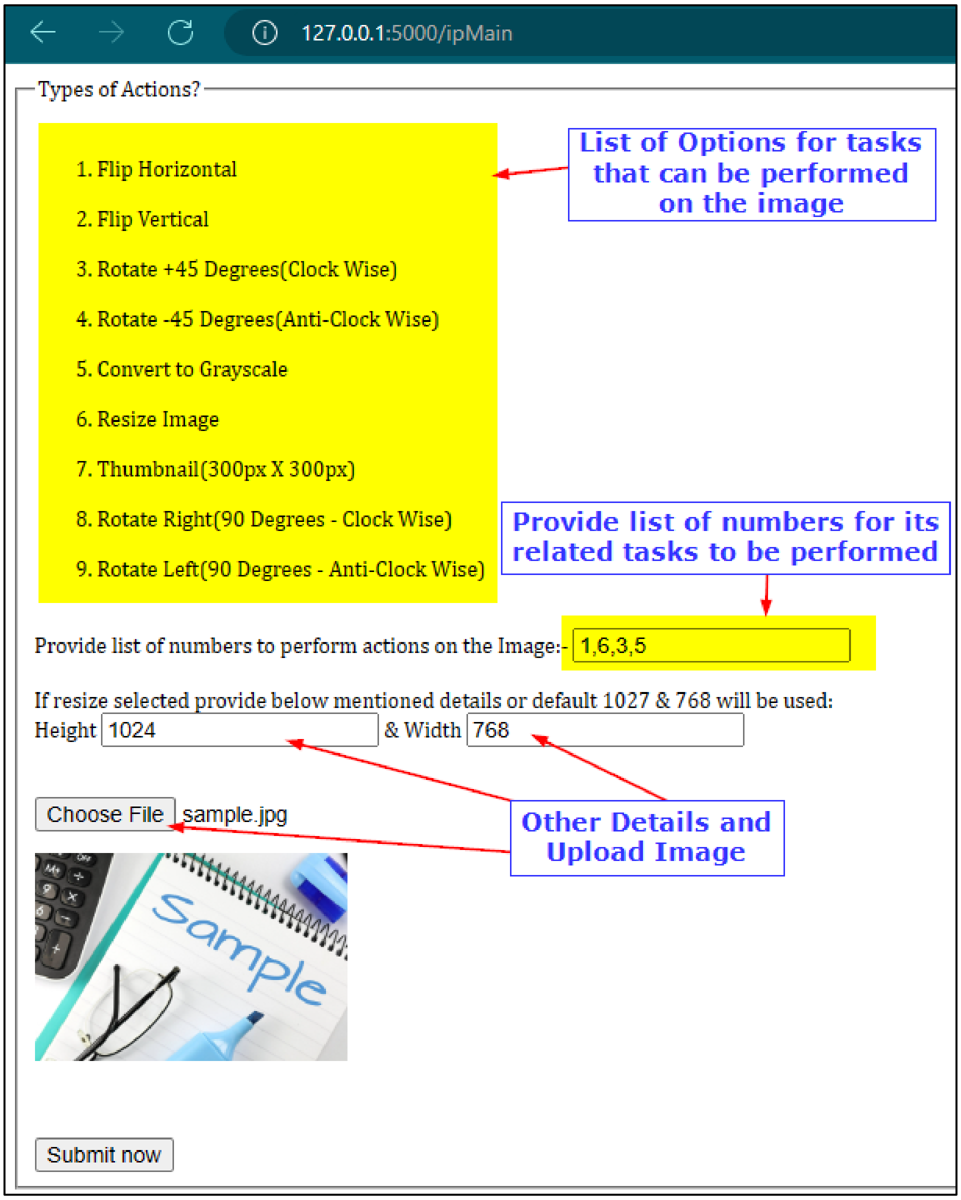
URL to load client UI - http://127.0.0.1:5000/ipMain

```
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
            <title>IP</title>
    </head>
            <body>
                <!-- Load external JavaScript -->
                <script src="{{url_for('static', filename='IP_script.js')}}"></script>
                <script>
                    var loadFile = function(event) {
                        var image = document.getElementById('output');
                        image.src = URL.createObjectURL(event.target.files[0]);
                    };
                    function CheckNumeric(e) {
                        if (window.event) // IE
                        {
                            if ((e.keyCode <48 || e.keyCode > 57) & e.keyCode != 8 && e.keyCode != 44) {
                                event.returnValue = false;
                                return false;
                            }
                        }
                        else { // Fire Fox
                            if ((e.which <48 || e.which > 57) & e.which != 8 && e.which != 44) {
                                e.preventDefault();
                                return false;
                            }
                        }
                    };
                </script>
            <form id="user_form" action = "/editImage" method = "POST" enctype = "multipart/form-data">
                    <fieldset>
                        <legend>Types of Actions?</legend>
                            <ol>
                                <br><li>Flip Horizontal</li>
                                <br><li>Flip Vertical</li>
                                <br><li>Rotate +45 Degrees(Clock Wise)</li>
                                <br><li>Rotate -45 Degrees(Anti-Clock Wise)</li>
                                <br><li>Convert to Grayscale</li>
                                <br><li>Resize Image</li>
                                <br><li>Thumbnail(300px X 300px)</li>
                                <br><li>Rotate Right(90 Degrees - Clock Wise)</li>
                                <br><li>Rotate Left(90 Degrees - Anti-Clock Wise)</li>
                            </ol>
                        <br>
                            <label for="ListOfActionsLabel">Provide list of numbers to perform actions on the
Image:-</label>
                              <input type="Text" id="ListOfActions" name="ListOfActions" placeholder="List Of
Actions" value="1,6,3,5" onkeypress="CheckNumeric(event);" required/>
                            </br>
                            <br>
                                   <label for="Resize">If resize selected provide below mentioned details or
default 1027 & 768 will be used:</label>
                                <br>
                                            Height <input type="Text" id="Height" name="Height" value="1024"
placeholder="Height" onkeypress="CheckNumeric(event);" required/>
                                             & Width <input type="Text" id="Width" name="Width" value="768"
placeholder="Width" onkeypress="CheckNumeric(event);" required/>
                                </br>
                            </br>
                            <br>
                                         <input type="file" id="imageFile" name="imageFile" accept="image/*"
onChange="loadFile(event)" required>
                                <p> <img id="output" width="200"/> </p>
                            </br>
                            <br>
                            <input value="Submit now" type="submit"/>
                        </br>
                        </fieldset>
                    </form>
                </body>
            </html>
```

Client-Side Code 2 – IP_script.js

```
/* Place your JavaScript in this file */
var loadFile = function(event) {
    var image = document.getElementById('output');
    image.src = URL.createObjectURL(event.target.files[0]);
};
var submitFunc = function() {
    let reqJSON = {}
    reqJSON.ListOfActions = document.getElementById('ListOfActions').value
    reqJSON.image = document.getElementById("Original_Image_File").files[0]
    console.log('reqJSON --', reqJSON)
    xhr.send(formData);
}
```

Insert Data as required :-



127.0.0.1:5000/ipMain

Types of Actions?

1. Flip Horizontal
2. Flip Vertical
3. Rotate +45 Degrees(Clock Wise)
4. Rotate -45 Degrees(Anti-Clock Wise)
5. Convert to Grayscale
6. Resize Image
7. Thumbnail(300px X 300px)
8. Rotate Right(90 Degrees - Clock Wise)
9. Rotate Left(90 Degrees - Anti-Clock Wise)

List of Options for tasks that can be performed on the image

Provide list of numbers for its related tasks to be performed

Provide list of numbers to perform actions on the Image:- 1,6,3,5

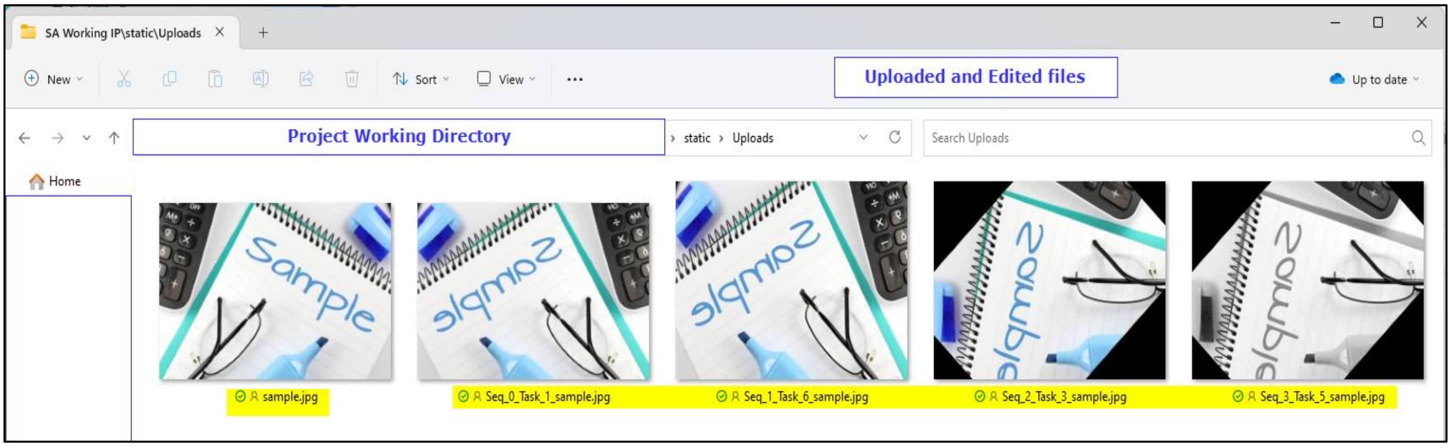
If resize selected provide below mentioned details or default 1027 & 768 will be used:
Height 1024 & Width 768

Choose File sample.jpg

Other Details and Upload Image

Submit now

Uploaded Images will appear here :-



SA Working IP\static\Uploads

New · Sort · View · ...

Uploaded and Edited files

Up to date

Project Working Directory › static › Uploads

Search Uploads

Home

sample.jpg

Seq_0_Task_1_sample.jpg

Seq_1_Task_6_sample.jpg

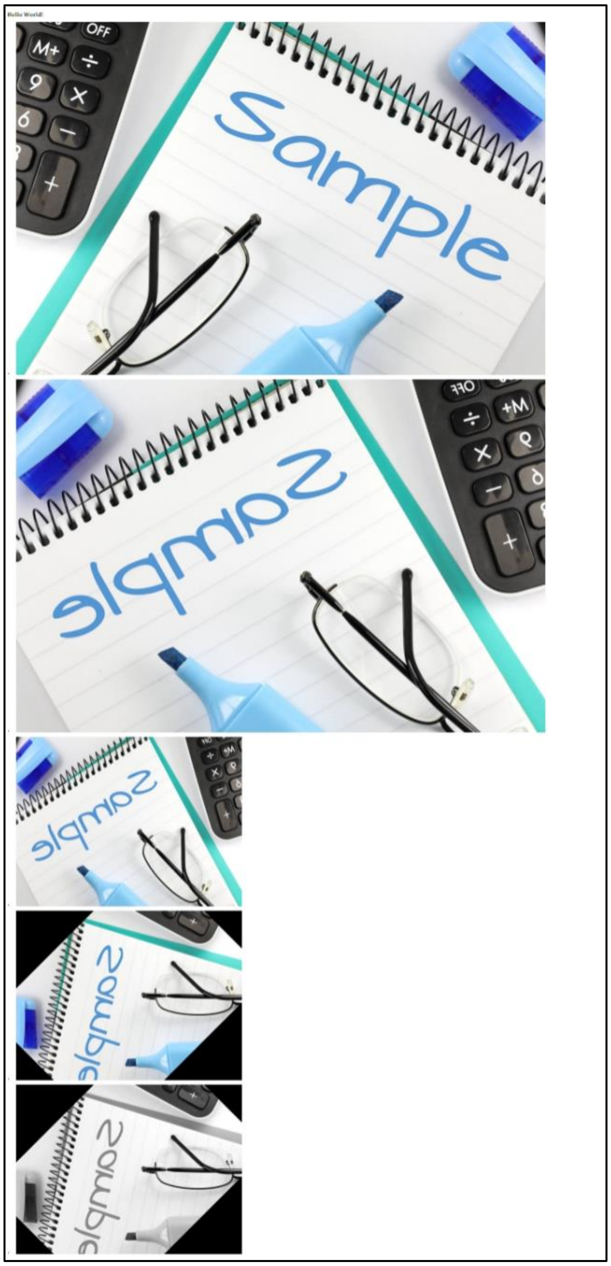
Seq_2_Task_3_sample.jpg

Seq_3_Task_5_sample.jpg

Output Redirection –

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Hello Worl!</title>
</head>
<body>
<h1>Hello World!</h1>
<ul>
{% for images in imageList %}
<li><img src = "{{ url_for('static', filename=images)}}"></li>
<br>
{% endfor %}
</body>
</html>
```

Output Page :-



Hello World!

Swagger
Supported by SMARTBEAR

/image_modifier.json Explore

Swagger UI document 1.0

`[ Base URL: 127.0.0.1:5000 ]`

/image_modifier.json

This document shows Swagger UI document for Image modification

Image Modifier API

GET `/modifyImage`

Parameters Try it out

Name	Description
rotateLeft *(path)*	Left rotation false
rotateRight *(path)*	Right rotation false
thumbnail *(path)*	Create thumbnail false

Responses **Response content type** application/json

Code	Description
200	Image Modified Successfully
400	Bad Request
500	Internal Server Error

[Powered by Flasgger 0.9.5]