Лабораторная работа по теме №10 Генерация изображения веб-сервером по запросу

Задание

Напишите корневой адрес работающего по https веб-приложения, которое по маршруту /makeimage получает значения width и height (выраженные в пикселах) в виде параметров POST-запроса из HTML формы страницы на сервере Flask (т.е. например width=100, height=200 и text=Hello) и возвращает изображение (в виде массива байтов) в формате JPG (image/jpg) с этими размерами, закодированное в кодировке base64 и отображаемое на другой HTML странице сервера Flask. Альтернативно можно предоставлять относительный публичный URL картинки, сохранённый на стороне сервера Flask как на видео по ссылке в начале задания. Возможно, следует отправлять заголовок, Content-Type со значением image/jpg. По маршруту /login приложение должно выдавать ваш логин в этой системе (MOODLE).

Требования: :small_red_triangle: Формат ответа по маршруту /login {"author": "**ваш логин**"} :small_red_triangle: Если переданы неверные параметры размеров изображения, возвращать сообщение в переменной message (см. видео) "Invalid image size" на ту же самую страницу makeimage.html снова с формой для генерации картинки. :small_red_triangle: Усложнения (дают: вероятность повышение итоговой оценки за модуль, прирост кармы и т.д.):

1. Реализация дополнительных решений более чем с 1 фреймворком (Варианты: Node.js, Django, Go)
2. Реализация фронтэнда и отправка данных на сервер с использованием какого-либо фронтэнд-фреймворка.
3. Реализация решения без фронтэнда, т. е. только бэкэнда (проверка с помощью Postman, учесть CORS).
4. Вывод и отображение самого изображения на страницу с формой (без перезагрузки страницы, асинхронно) или другую страницу как thumbnail.

Код программы

```
from flask import Flask, request, jsonify, render_template, send_file
from flask_cors import CORS
from PIL import Image, ImageDraw
import io
import base64
import os

app = Flask(__name__)
CORS(app)

@app.route('/')
def home():
  return "Flask Image Generator is Running!"

@app.route('/login', methods=['GET'])
def login():
```

```
  return jsonify({"author": "1149284"})

@app.route('/makeimage', methods=['GET', 'POST'])
def make_image():
  if request.method == 'GET':
      return render_template('makeimage.html')
  try:
      width = int(request.form.get('width'))
      height = int(request.form.get('height'))
      text = request.form.get('text', '')

      if width <= 0 or height <= 0:
          return render_template('makeimage.html', message="Invalid image
size")

      img = Image.new('RGB', (width, height), color=(73, 109, 137))
      d = ImageDraw.Draw(img)
      d.text((10, 10), text, fill=(255, 255, 0))

      img_byte_arr = io.BytesIO()
      img.save(img_byte_arr, format='JPEG')
      img_byte_arr.seek(0)

      return send_file(img_byte_arr, mimetype='image/jpeg')


  except (ValueError, TypeError):
      return render_template('makeimage.html', message="Invalid image size")

if __name__ == '__main__':
  app.run(host='0.0.0.0', port=8080)  # Убираем ssl_context для Replit
```

Результат