

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования «Томский государственный университет систем
управления и радиотехники»

Факультет систем управления (ФСУ)

Кафедра автоматизированных систем управления (АСУ)

&lab_name
отчет по лабораторной работе №& по дисциплине
«&course_name»

Обучающийся гр. 431-3
_____ Сергиевский Д.В.
«&__» & _____ 202& г.

Проверил: доцент каф. АСУ, к.т.н.
_____ &Фамилия И.О.
«&__» & _____ 202& г.

Томск 202&

Содержание

Введение.....	3
1 Задание из методического пособия.....	4
2 Задание по варианту.....	6
2.1 Постановка задачи.....	6
2.2 Структура проекта.....	8
2.3 Описание работы приложения.....	10
3 Вывод.....	13
Приложение А.....	14

Введение

В рамках данной лабораторной работы требуется ознакомиться с параграфом 2.13 методического пособия, ознакомиться с фреймворком Flask и разработать небольшое веб-приложение.

Цели:

- Ознакомиться с фреймворком Flask.

Задания:

- Повторить действия, изложенные в методическом пособии.
- Разработать веб-приложение в соответствии с заданием по варианту.
- Выполнить индивидуальное задание при получении оного.

Задание по варианту (18):

- Веб-приложение должно формировать новое изображение на основе исходного путем умножения изображения на периодическую функцию \sin или \cos с нормировкой, период изменения задает пользователь, аргумент функции определяется вертикальной или горизонтальной составляющей. Нарисовать график распределения цветов для нового и исходного изображения. Каждое приложение должно обеспечивать проверку на работа с помощью капчи или любой другой технологии. Разместите объекты отображения и ввода удобно для пользователя.

Индивидуальное задание:

-

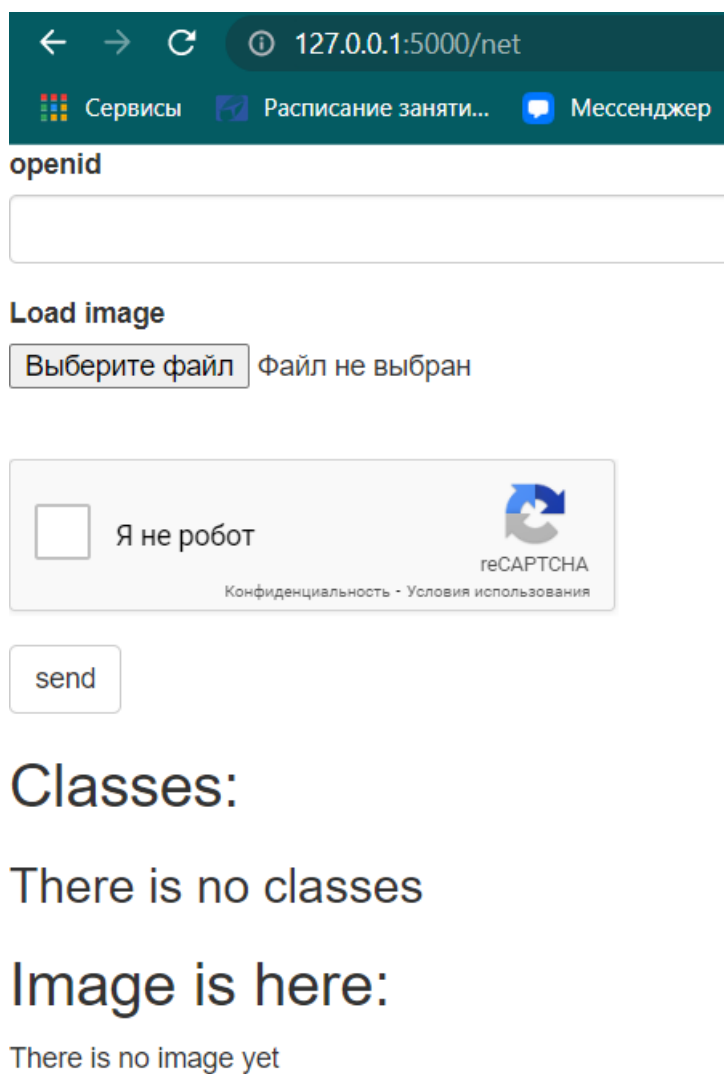
1 Задание из методического пособия

Задание: выполнить изложенные в методическом пособии примеры для ознакомления с основами разработки веб-приложений с использованием Flask.

В ходе данного задания был разобран пример из методического пособия — веб-приложение, производящее некую классификацию загружаемого изображения.

Пример работы приложения представлен на Рисунках 1-2.

Исходные файлы приложения были помещены в директорию `manual_app`, размещенную в корневой директории проекта. Исходный код представлен в Приложении А.



The screenshot shows a web browser window with the address bar displaying `127.0.0.1:5000/net`. Below the address bar, there are three links: "Сервисы", "Расписание заняти...", and "Мессенджер". The main content area contains the following elements:

- A label `openId` followed by an empty text input field.
- A label `Load image` followed by a button labeled "Выберите файл" and the text "Файл не выбран".
- A reCAPTCHA widget with a checkbox labeled "Я не робот" and the reCAPTCHA logo. Below the checkbox, there is a link for "Конфиденциальность - Условия использования".
- A button labeled `send`.
- A heading `Classes:` followed by the text `There is no classes`.
- A heading `Image is here:` followed by the text `There is no image yet`.

Рисунок 1.1 - основная страница

←

→

↻

127.0.0.1:5000/net

Сервисы

Расписание заняти...

Мессен

openid

123


Load image

Выберите файл

Файл не выбран

☐

Я не робот



reCAPTCHA

Конфиденциальность · Условия использования

send

Classes:

brass: 0.98176736

menu: 0.99035114

Image is here:

./static\pynames.jpg

Type	Public	Internal
Packages	lower_sith_under	

Рисунок 1.2 - основная страница после загрузки изображения

2 Задание по варианту

2.1 Постановка задачи

Задание (вариант 18): Веб-приложение должно формировать новое изображение на основе исходного путем умножения изображения на периодическую функцию \sin или \cos с нормировкой, период изменения задает пользователь, аргумент функции определяется вертикальной или горизонтальной составляющей. Нарисовать график распределения цветов для нового и исходного изображения. Каждое приложение должно обеспечивать проверку на робота с помощью капчи или любой другой технологии. Разместите объекты отображения и ввода удобно для пользователя.

Данное задание было решено выполнить с использованием двух страниц веб-приложения, посвященных выполнению соответствующих подзадач.

На первой странице должна происходить загрузка изображения с проверкой корректности вводимых данных и последующей переадресацией на вторую страницу. Также на данную страницу была вынесена проверка на робота с помощью ReCAPTCHA. Данной странице был присвоен адрес `"/upload"`.

На второй странице должно происходить формирование нового изображения путем преобразование исходного. Данной странице был присвоен адрес `"/retouch/{image_name}"`, где `image_name` — имя исходного изображения

Пользователю должны быть доступны для ввода следующие параметры:

- функция, участвующая в формировании нового изображения: синус или косинус;
- составляющая, выступающая в качестве переменного аргумента функции: горизонтальная или вертикальная координата пикселя;
- период функции, выраженный в пикселях или процентах от максимального значения выбранной составляющей по выбору.
- После ввода корректных данных на странице должны отобразиться сформированное изображение и графики распределения цветов для обоих изображений.
- В качестве графика распределения цветов была принята гистограмма изображения, разделенная по цветовым составляющим.

В связи с тем, что однозначно подобрать нормировку, сохраняющую особенности изображения, не удалось, вместо умножения изображения на значение функции и нормировки было выбрано преобразование при котором в точках, соответствующих впадинам функции, изображение затемняется, а в точках, соответствующих гребням, осветляется. Таким образом, при значении функции 0 соответствующий пиксель не меняет цвет, при значениях 1 и -1 цвет обращается в белый и черный соответственно. Поскольку данное условие поставлено исключительно с целью проверки навыков работы с изображениями и внесения уникальности в варианты задания, подобное нарушение показалось приемлемым.

В процессе постановки и решении задачи были выявлены некоторые проблемы, связанные с выбранной архитектурой приложения.

Во первых, изображение доступно по своему имени, что позволяет любому пользователю получить к нему доступ и делает невозможным параллельную работу пользователей над одноименными изображениями

Во вторых, после работы пользователя с изображением сохраняются остаточные файлы, которые никогда не будут удалены.

Главным образом данные проблемы связаны с решением разделить загрузку и обработку изображения на отдельные страницы с целью декомпозировать задачу. Без изменения выбранной архитектуры решением могло стать идентификация пользователей для разделения их рабочих областей и отслеживание окончания сессии, тем не менее очевидных инструментов, реализующих данные возможности, найдено не было.

Поскольку данное приложение было выполнено с ознакомительными целями и в связи с отсутствием требований к работе приложения при участии нескольких пользователей, данные проблемы было решено проигнорировать.

2.2 Структура проекта

В ходе работы проект принял структуру, представленную на Рисунке 1.

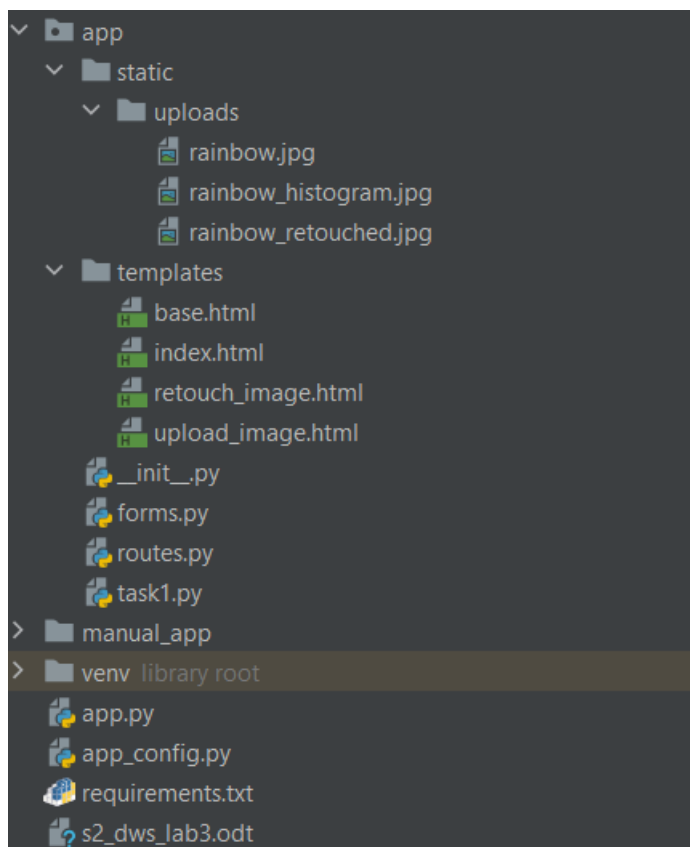


Рисунок 2.2.1 - структура проекта

В директории `manual_app` размещены исходные файлы веб-приложения, представленного в методическом пособии.

Файл `app.py` используется для запуска веб-приложения, `app_config.py` содержит константы, регулирующие работу приложения. Исходные файлы веб-приложения размещены в директории `app`.

В файле `routes.py` описаны функции, реализующие обработку запросов. Маршрутизация выполнена стандартными возможностями Flask. Для работы с загружаемыми изображениями использован модуль `PIL.Image`, для приведения имен к корректному виду использовалась функция `secure_filename` модуля `werkzeug.utils`.

В файле `forms.py` описаны формы, созданные с помощью стандартных модулей `flask_wtf` и `wtforms`. Соответствующие шаблоны размещены в папке `templates`.

В файле `task1.py` реализованы функции, выполняющие обработку в соответствии с заданием по варианту. Обработка выполнялась с помощью модулей `PIL.Image` и `numpy`, получение графиков — с помощью модуля `matplotlib`.

Директория `static/uploads` используется для хранения загруженных изображений и производных от них.

Исходный код приведен в Приложении Б.

2.3 Описание работы приложения

Приложение поддерживает обращение по следующим адресам:

- /index (/): корневая страница, не содержащая контента и оставленная исключительно в целях заполнения места
- /upload: страница, выполняющая загрузку изображения. Скриншот представлен на Рисунке 1.
- /retouch/<image_name>: страница, выполняющая обработку изображения. Изображение идентифицируется через параметр image_name. Скриншот представлен на Рисунке 2.
- /retouch/default: производит переадресацию на страницу обработки с использованием изображения по умолчанию.

Помимо необходимых для работы компонентов сверху каждой страницы расположена панель с ссылками на основные страницы для удобства переходов.

Результаты обработки изображения представлены на Рисунке 3.

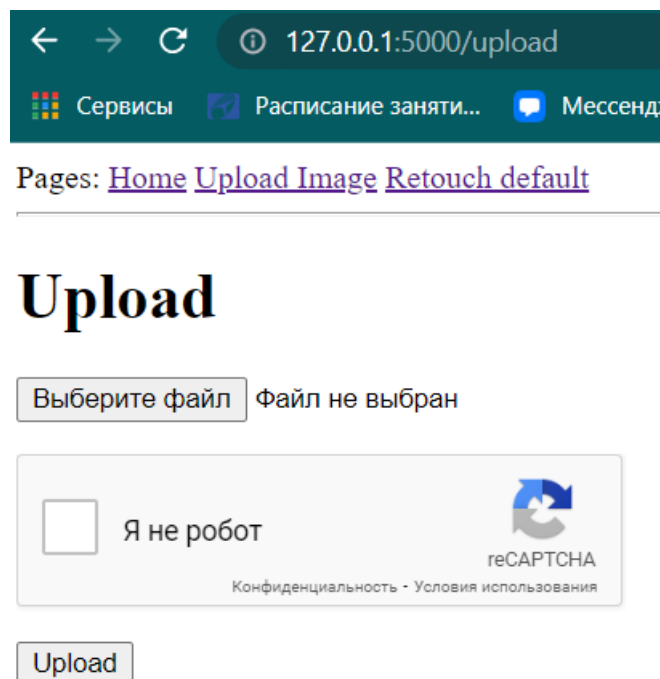


Рисунок 2.3.1 - страница загрузки

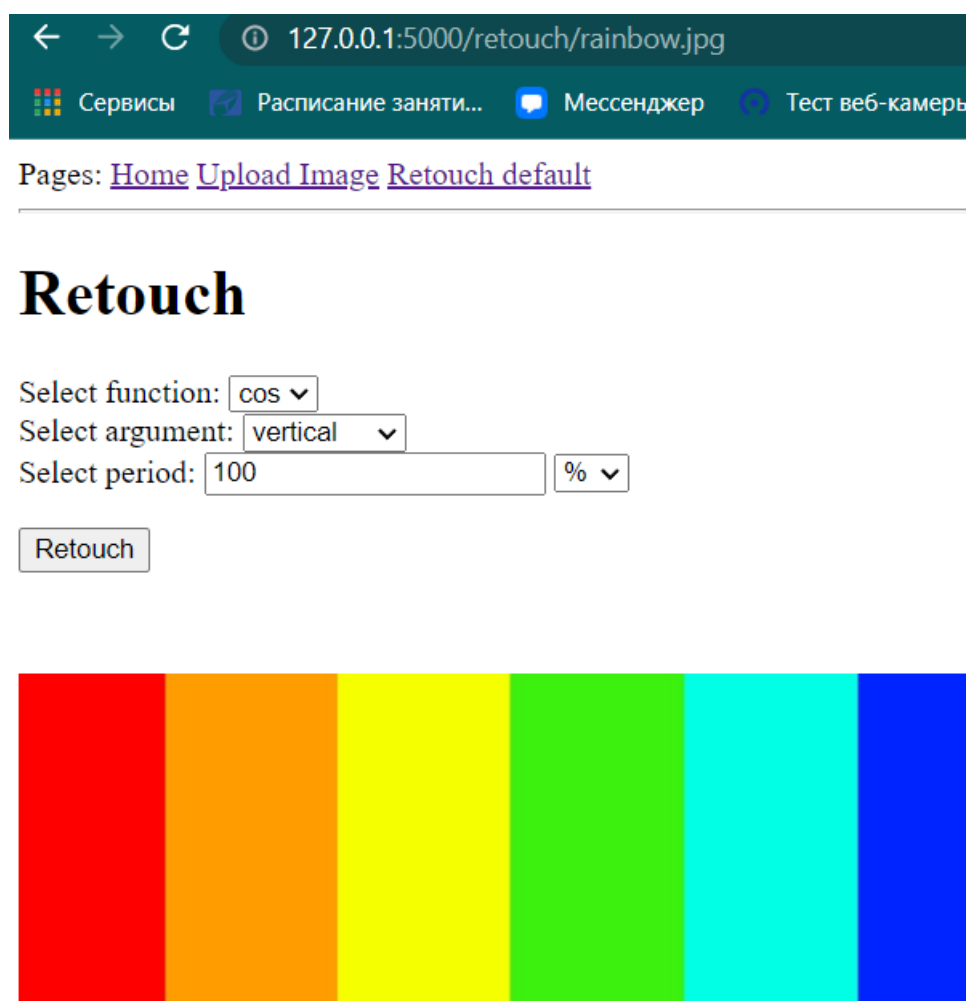


Рисунок 2.3.2 - страница обработки

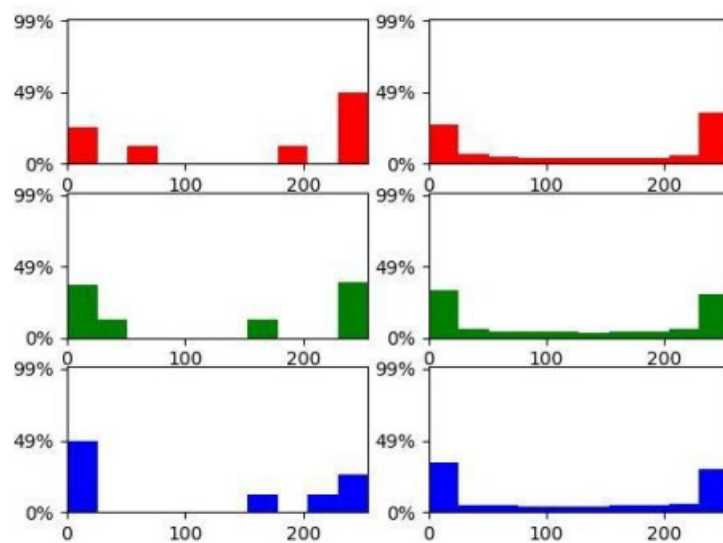
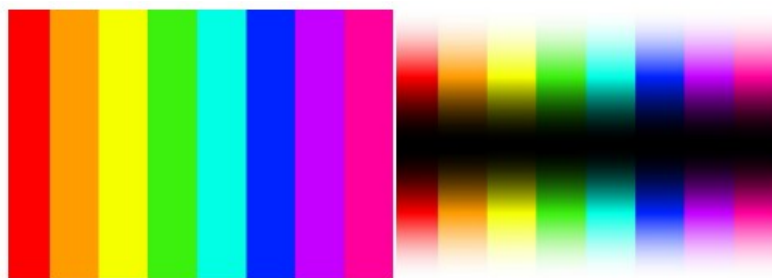


Рисунок 2.3.3 - результат обработки

3 Вывод

В ходе выполнения данной лабораторной работы было проведено знакомство с фреймворком Flask и получены базовые навыки разработки веб-приложений.

Также полученные навыки были применены на практике в ходе выполнения заданий.

Приложение А

Исходный код веб-приложения, созданного в соответствии с заданиями методического пособия представлен по ссылке github.com/serdenv1/s2_dws_lab3 в директории `manual_app`.

Приложение Б

Исходный код веб-приложения, созданного в соответствии с заданием по варианту\ представлен по ссылке github.com/serdenvl/s2_dws_lab3.