Системы обработки изображений в медицине

Климочкина
Ксения Б19-302

Постановка задачи

Основной задачей моего проекта является отправка изображений на отладочный веб-сервер Django с графическими примитивами

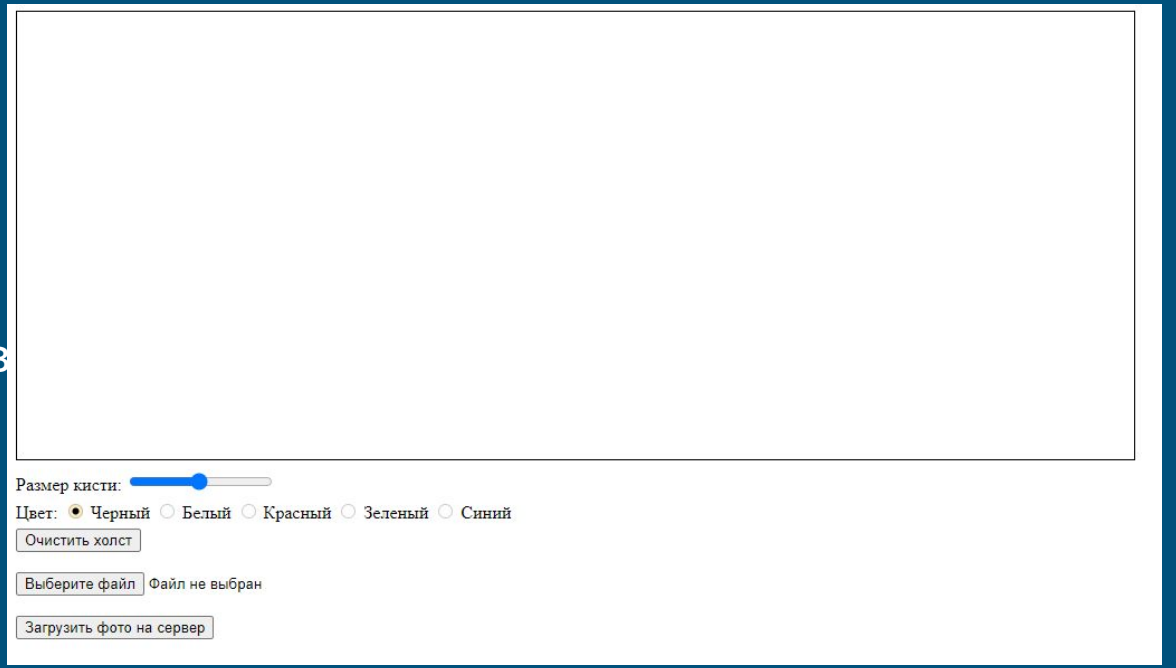
Использованные технологии

- HTML - интерфейс сайта
- JavaScript - отрисовка графических примитивов
- Django - развертывание веб-сервера

Интерфейс

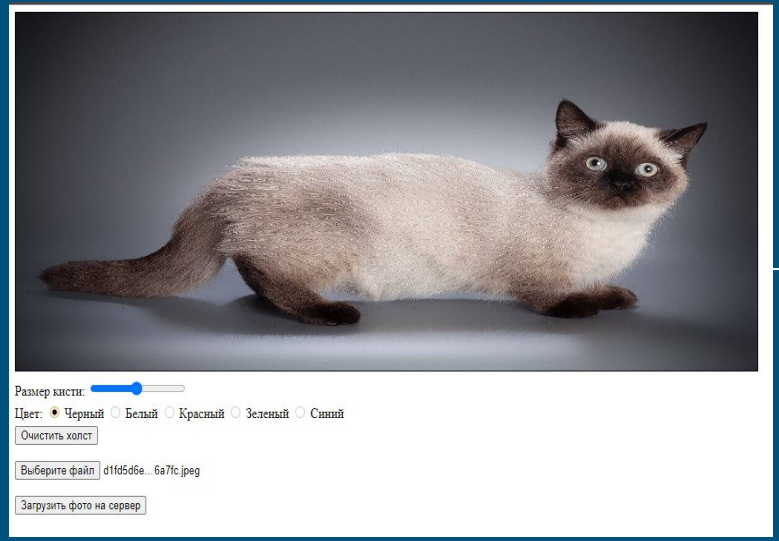
Основные компонуемые:

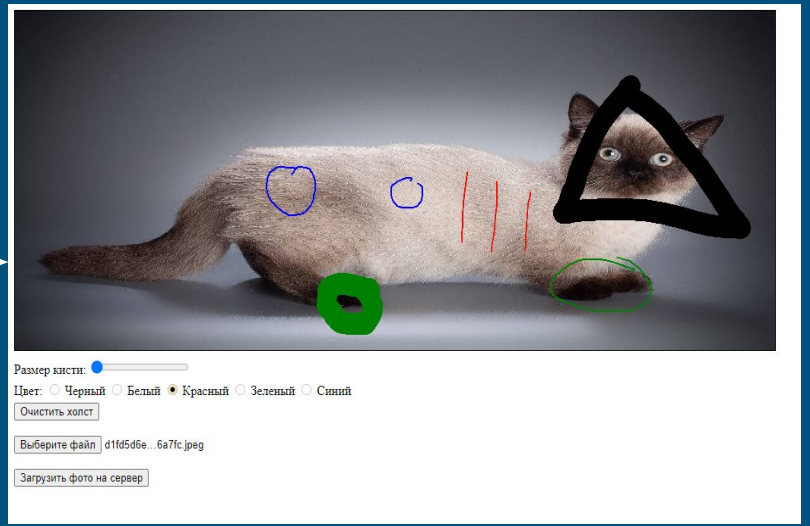
- Кисть для нанесения графических примитивов
- Кнопка очистки нанесенных примитивов
- Кнопка загрузки изображения
- Кнопка загрузки изображения на сервер



Работа программы

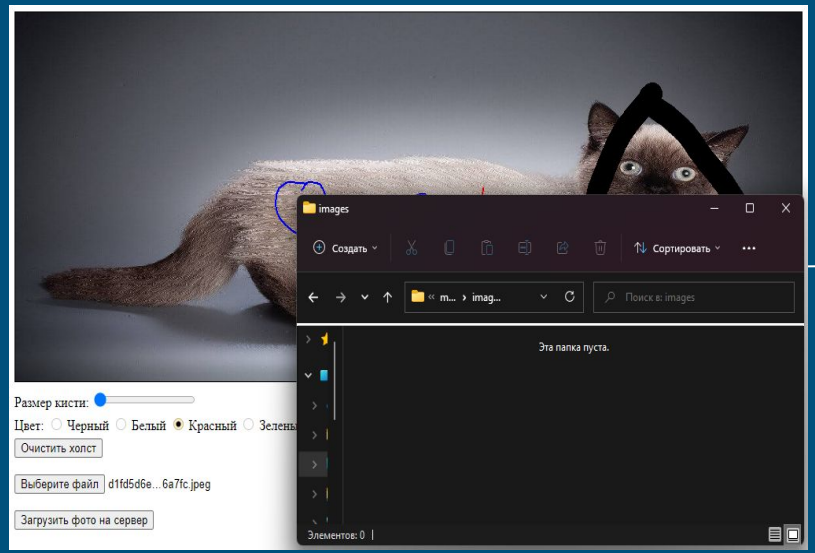
Нанесение графических примитивов различных цветов и размеров

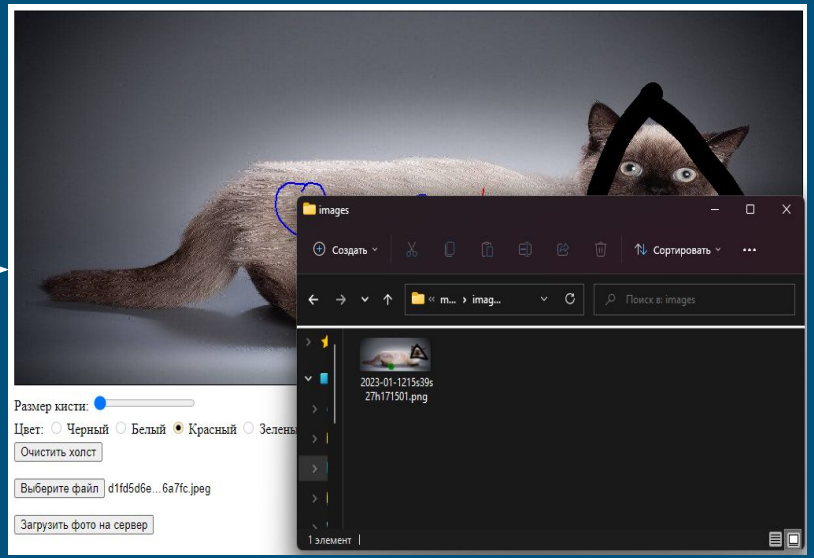




Работа программы

Сохранение нового изображения





Настройки settings.py

Добавлено приложение ‘page’ для работы решения

Подключена папка ‘static’ для использования JavaScript

Подключена папка ‘media’ для сохранения изображений

```
INSTALLED_APPS = [
…
    'page',
]

STATICFILES_DIRS = [
   os.path.join(BASE_DIR, "static"),
]
STATIC_URL = 'static/'


MEDIA_URL = "/media/"
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
```

Настройка urls.py

Настроена маршрутизация по сайту

```
from django.conf import settings
from django.conf.urls.static
import static
from django.contrib import admin
from django.urls import path,
include

urlpatterns = [
   path('admin/',
admin.site.urls),
   path('',include('page.urls'))
] + static(settings.MEDIA_URL,
document_root=settings.MEDIA_ROO
T)
```

Маршрутизация

Настроена маршрутизация в самом приложении

Используются два адреса:

‘page’ используется для отображения интерфейса приложения

‘save pic’ используется для сохранения изображения на сервере

```
from django.urls import path
from .views import save_pic,page

urlpatterns = [
    path('page/', page,
name="images"),
    path('save_pic/', save_pic,
name='save_pic'),
]
```

Файл views.py

Функция ‘save pic’ принимает изображение через POST запрос в формате dataURL и декодирует его в изображение и сохраняет в отдельную папку с индивидуальным названием .

Функция ‘page’ отображает page.html

```
from django.shortcuts import render
from django.http import HttpResponse
from django.views.decorators.csrf import csrf_exempt
from base64 import b64decode
import datetime

@csrf_exempt
def save_pic(request):
    if request.method == 'POST':
        data_uri = request.POST['p']
        header, encoded = data_uri.split(",", 1)
        data = b64decode(encoded)
        name =
"C:/h3/media/images/"+str(datetime.datetime.now()).rep
lace(" ","").replace(".","h").replace(":","s")+".png"
        with open(name, "wb") as f:
            f.write(data)
    return HttpResponse("GOOD!", status=200)

def page(request):
    return render(request, 'files/page.html')
```

Файл page.html

Архитектура файла

Программа создает полотно, на котором прорисовываются графические примитивы и на которое загружается изображение.

Далее представлено описание настройки кисти и добавление функциональных кнопок.

```
<!DOCTYPE html>
<html>
<script
src="//ajax.googleapis.com/ajax/libs/jquery/1.9.1/jquery.min.js"   ></script>
 <body>
   <div>
     <canvas
       id="canvas"
       width="1000"
       height="400"
       style="border: 1px solid black ;"
     ></canvas>
   </div>
   <div style="margin-top : 5px;">
     <span> Размер кисти:  </span>
     <input
       type="range"
       min="1"
       max="50"
       value="25"
       class="size"
       id="sizeRange"
     />
   </div>
   <div style="margin-top : 5px;">
     <span> Цвет: </span>
     <input type="radio" name="colorRadio" value="black" checked />
     <label for="black">Черный</label>
     <input type="radio" name="colorRadio" value="white" />
     <label for="black">Белый</label>
     <input type="radio" name="colorRadio" value="red" />
     <label for="black">Красный</label>
     <input type="radio" name="colorRadio" value="green" />
     <label for="black">Зеленый</label>
     <input type="radio" name="colorRadio" value="blue" />
     <label for="black">Синий</label>
   </div>
   <div style="margin-top : 5px;">
     <button id="clear">Очистить холст </button>
   </div>
     <br/>
   <input id="upload" type="file" accept="image/*" />
   <br/>
   <form method="get">
       <br/>
     <button type="button" onClick="save()">Загрузить фото на
сервер</button>
       <br/>
   </form>
   {% load static %}
   <script src="{% static '/draw.js' %}"></script>
   {% block content %}{% endblock %}
 </body>
</html>
```

Файл draw.js

В этой файле находятся функции отрисовки и загрузки изображения.

На слайд вынесена часть кода, который отвечает за отправку изображения с графическими примитивами на веб - сервер.

```
function save(){
 const
canvas=document.getElementById('canvas');
 const  imageUri = canvas.toDataURL();
 console.log(imageUri);
 $.ajax({
   type: 'POST',
   url: '/save_pic/',
   data: {
     'p': imageUri
   },
   success: function(response){
     console.log(response)
   },
   error: function(response){
     console.log(response)
   }
 })
}
```

СПАСИБО ЗА ВНИМАНИЕ !!

Ссылка на - > github