МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Допущено к защите

Руководитель проекта

\_\_\_\_\_\_\_\_\_ (Исупов К. С.)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2022г.

Разработка онлайн фоторедактора

Пояснительная записка курсового проекта по дисциплине

«Комплекс знаний бакалавра в области программного и аппаратного обеспечения вычислительной техники»

ТПЖА.09.03.01.487 ПЗ

Разработал студент группы ИВТ-21 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Птахова А.М./

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Исупов К. С./

Консультант \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Кошкин О. В./

Проект защищен с оценкой «\_\_\_\_\_\_\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(оценка) (дата)*

Члены комиссии \_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_/\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись)

Киров 2022

**Реферат**

Птахова А. М. Разработка онлайн фоторедактора: ТПЖА.090301.487ПЗ: Курс. проект / ВятГУ, каф. ЭВМ; рук. Исупов К. С. - Киров, 2022. – ПЗ 45 с, 11 рис., 1 табл., 4 источника.

Объект исследования и разработки – программный продукт онлайн фоторедактор

Цель курсового проекта – разработать веб приложение “фоторедактор”.

Результатом выполнения курсового проекта является прототип программного обеспечения и проектная документация.

Разработанное программное обеспечение реализует функции, необходимые для предоставления пользователю возможности редактирования фотографий.

Оглавление

[Введение 4](#_Toc100509332)

[1. Анализ предметной области 5](#_Toc100509333)

[1.1 Актуальность темы 5](#_Toc100509334)

[1.2 Обзор аналогов 5](#_Toc100509335)

[1.2.1 Fotor.com 5](#_Toc100509336)

[1.2.2 Pixlr.com 5](#_Toc100509337)

[2. Постановка задачи 7](#_Toc100509338)

[2.1.2 Требования к функциям 7](#_Toc100509339)

[2.1.2. Требования к интерфейсу 7](#_Toc100509340)

[2.1.3 требования к входным и выходным данным 7](#_Toc100509341)

[2.1.4 требования к ПО 7](#_Toc100509342)

[3. Разработка структуры программы 8](#_Toc100509343)

[3.1 Архитектура приложения 8](#_Toc100509344)

[3.2Функциональное взаимодействие блоков 8](#_Toc100509345)

[Обозначенные выше функции можно разбить на следующие блоки 8](#_Toc100509346)

[3.3 Разработка структуры базы данных 10](#_Toc100509347)

[3.4 Разработка алгоритмов 11](#_Toc100509348)

[3.5 Разработка интерфейса пользователя 12](#_Toc100509349)

[4. Программная реализация 13](#_Toc100509350)

[5. Вывод 14](#_Toc100509351)

[Библиографический список 15](#_Toc100509352)

[Приложение А 16](#_Toc100509353)

[Приложение Б 19](#_Toc100509354)

[Приложение В 24](#_Toc100509355)

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | | | | | |
|  |  |  |  |  |
| Изм. | Лист | № докум | Подпись | Дата |
| Разраб | | Птахова А. М |  |  | Онлайн фоторедактор | Литера | | | Лист | Листов |
| Пров | | Исупов К. С |  |  |  |  |  | 3 | 45 |
|  | |  |  |  |  | | | | |
|  | |  |  |  |
| Реценз. | |  |  |  |

# Введение

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 4 |
| Изм. | Лист | № докум | Подпись | Дата |

В настоящее время создано много приложений, позволяющих пользователям выкладывать свои фотографии в открытый доступ, чтобы их могли заметить как можно больше людей. В связи с этим возникает желание улучшить свою фотографию при помощи фоторедактора.

Таких приложений достаточно много и они широко используются для редактирования. К сожалению, функционал некоторых очень сильно перегружен, что усложняет его использование начинающему фоторедактору. Поэтому было принято решение разработать веб приложение, которое содержит в себе основные функции для редактирования фотографий.

# 1. Анализ предметной области

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 5 |
| Изм. | Лист | № докум | Подпись | Дата |

# 1.1 Актуальность темы

В настоящее время фоторедактор – незаменимый инструмент при работе с фотографией.

Огромное количество изображений перед тем, как были опубликованы в общественном доступе, претерпели изменения при помощи редактирования.

# 1.2 Обзор аналогов

# 1.2.1 Fotor.com

Плюсы:

- интуитивно понятный интерфейс, представлен надписями

- есть всплывающие окна с объяснениями, как пользоваться

Минусы:

- большое число функций

- неудобное расположение панели управления

- панель управления занимает больше меньше, чем фото

# 1.2.2 Pixlr.com

Плюсы:

- удобная панель редактирования

- фотографии отводится больше всего места

Минусы:

- интерфейс представлен значками

- нет объяснений в использовании

|  |  |  |
| --- | --- | --- |
|  | Fotor.com | Pixlr.com |
| Понятный интерфейс | + | - |
| Объяснения | + | - |
| Фото занимает центральную позицию | - | + |
| Удобное расположение панели | - | + |

Таблица 1 – сравнение аналогов

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 6 |
| Изм. | Лист | № докум | Подпись | Дата |

В результате сравнения аналогов было принято решение разработать программу, включающее преимущества рассмотренных приложений.

# 2. Постановка задачи

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 7 |
| Изм. | Лист | № докум | Подпись | Дата |

Разработать веб приложение «фоторедактор» для оптимальной организации своей работы с изображениями

# 2.1.2 Требования к функциям

В программе должны быть реализованы следующие функции

- открытие изображения

- сжатие и расширение изображения

- поворот вправо и влево на 90

- фильтры

- возможность скачать изображения

# 2.1.2. Требования к интерфейсу

- расположение изображения по центру

- панель редактирования представлена надписями

# 2.1.3 требования к входным и выходным данным

Пользователь выбирает функцию редактирования, фотография обрабатывается на стороне сервера и отображается в измененном виде у пользователя.

# 2.1.4 требования к ПО

- операционная система Windows 7 и выше

# 3. Разработка структуры программы

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 8 |
| Изм. | Лист | № докум | Подпись | Дата |

# 3.1 Архитектура приложения

Приложение реализует клиент-серверную архитектуру. Принцип работы представлен на рисунке 1.

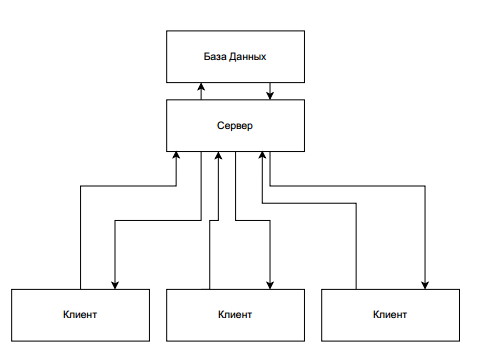


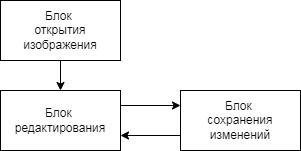
Рисунок 1 – клиент- серверная архитектура

# 3.2Функциональное взаимодействие блоков

# Обозначенные выше функции можно разбить на следующие блоки

1. Блок открытия изображения
2. Блок редактирования
3. Блок сохранения изменений

Схема взаимодействия функциональных блоков представлена на рисунке 2.



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 9 |
| Изм. | Лист | № докум | Подпись | Дата |

Рисунок 2 – Схема взаимодействия функциональных блоков

# 3.3 Разработка структуры базы данных

Большинство баз данных имеют табличную структуру. Таблицы – основные объекты базы данных. В таблицах хранятся необработанные данные. Каждая таблица содержит записи определенного вида. Запись – полный набор данных об определенном объекте. В режиме таблицы запись изображается как строка. Каждая запись таблицы содержит всю необходимую информацию об отдельном элементе базы данных.В режиме таблицы для представления поля используется столбец или ячейка, в этом случае имя поля является заголовком столбца таблицы.

Таким образом, поле – это наименьший поименованный элемент информации, хранящийся в базе данных и рассматриваемый как единое целое. Структура базы данных – это набор полей, которые определяют содержание и вид БД.

Для разработки курсового проекта была выбрана реляционная модель базы данных, совокупность связанных между собой двумерных таблиц, в которых хранится информация об объектах.

Схема структуры базы данных указана на рисунке 3.

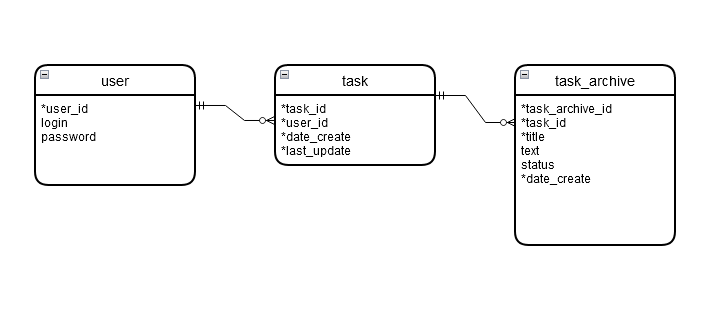


Рисунок 3 – схема структуры базы данных

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 10 |
| Изм. | Лист | № докум | Подпись | Дата |

# 3.4 Разработка алгоритмов

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 11 |
| Изм. | Лист | № докум | Подпись | Дата |

Были реализованы следующие алгоритмы для взаимодействия с фотографиями

- открытия изображения для редактирования

- поворот на 90 градусов вправо

- поворот на 90 градусов влево

- уменьшить изображение на: 25%, 50% и 75% от оригинала

- увеличить изображение на: 25%, 50% и 75% от оригинала

- фильтры (только для фотографий формата jpg)

- blur

- counter

- detail

- sharpen

- smooth

- сохранение изменений

- загрузка изображение (скачивание)

# 3.5 Разработка интерфейса пользователя

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 12 |
| Изм. | Лист | № докум | Подпись | Дата |

Интерфейс пользователя должен решать поставленную задачу, быть интуитивно понятным, не перегружен лишними элементами, а процесс взаимодействия пользователя с ним предсказуемым. Пользовательский интерфейс должен предоставлять доступ ко всему перечню функциональных возможностей, предусмотренных приложением.

С учетом данных принципов было принято решение сделать панель редактирования в виде выпадающего меню, разместить его горизонтально вверху страницы, а редактируемую фотографию отображать по центру страницы и под панелью редактированию.

# 4. Программная реализация

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 13 |
| Изм. | Лист | № докум | Подпись | Дата |

Для обработки изображений был выбрана библиотека Pillow, имеющая большой выбор простых инструментов для работы с изображениями. Так как библиотека Pillow подразумевает работу на языке Python, то он и был выбран для реализации функций редактирования.

Учитывая, что основной функционал будут написан на Python, было принято решение использовать фреймворк Django для создания веб приложения.

Для создания веб сервера был выбран Nginx, потому что работает быстрее при отдаче статики и потребляет меньше ресурсов сервера.

СУБД, которую Django предоставляет по умолчанию, является SQLite3. Этой базы данных вполне хватает для требований проекта, поэтому она и была использована.

# 5. Вывод

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 14 |
| Изм. | Лист | № докум | Подпись | Дата |

В ходе выполнения лабораторной работы были разработано веб приложение «фоторедактор», позволяющее пользователю сделать простое редактирование проще и удобнее.

# Библиографический список

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 15 |
| Изм. | Лист | № докум | Подпись | Дата |

1. Документация по nginx [Элетронный курс] – режим доступа <https://nginx.org/ru/docs>
2. Документация по Pillow [Электронный курс] – режим доступа <https://pillow.readthedocs.io/en/stable/handbook/tutorial.html>
3. Документация по html [Электронный курс] – режим доступа <http://htmlbook.ru/html>
4. Документация по django [Электронный курс] – режим доступа <https://docs.djangoproject.com/en/4.0>

# Приложение А

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 16 |
| Изм. | Лист | № докум | Подпись | Дата |

(Обязательное)

Алгоритмы

Схема алгоритма редактирования

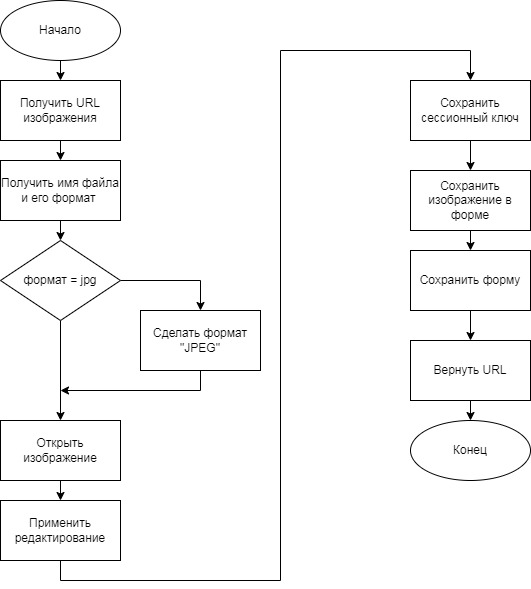


Рисунок 4 – схема алгоритма редактирования

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 17 |
| Изм. | Лист | № докум | Подпись | Дата |

Схема алгоритма открытия изображения

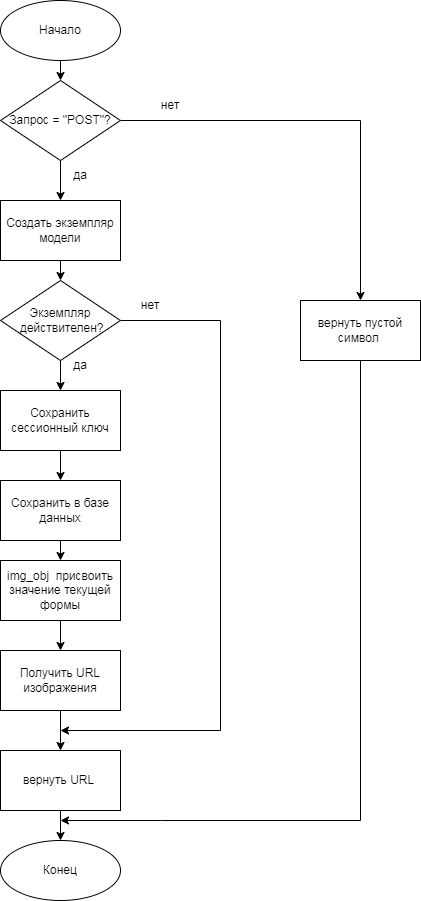


Рисунок 5 – схема алгоритма открытия изображения

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 18 |
| Изм. | Лист | № докум | Подпись | Дата |

Схема алгоритма сохранения изменений

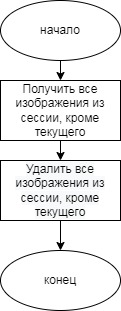


Рисунок 6 – схема алгоритма сохранения изменений

# Приложение Б

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 19 |
| Изм. | Лист | № докум | Подпись | Дата |

(Обязательное)

Экранные формы

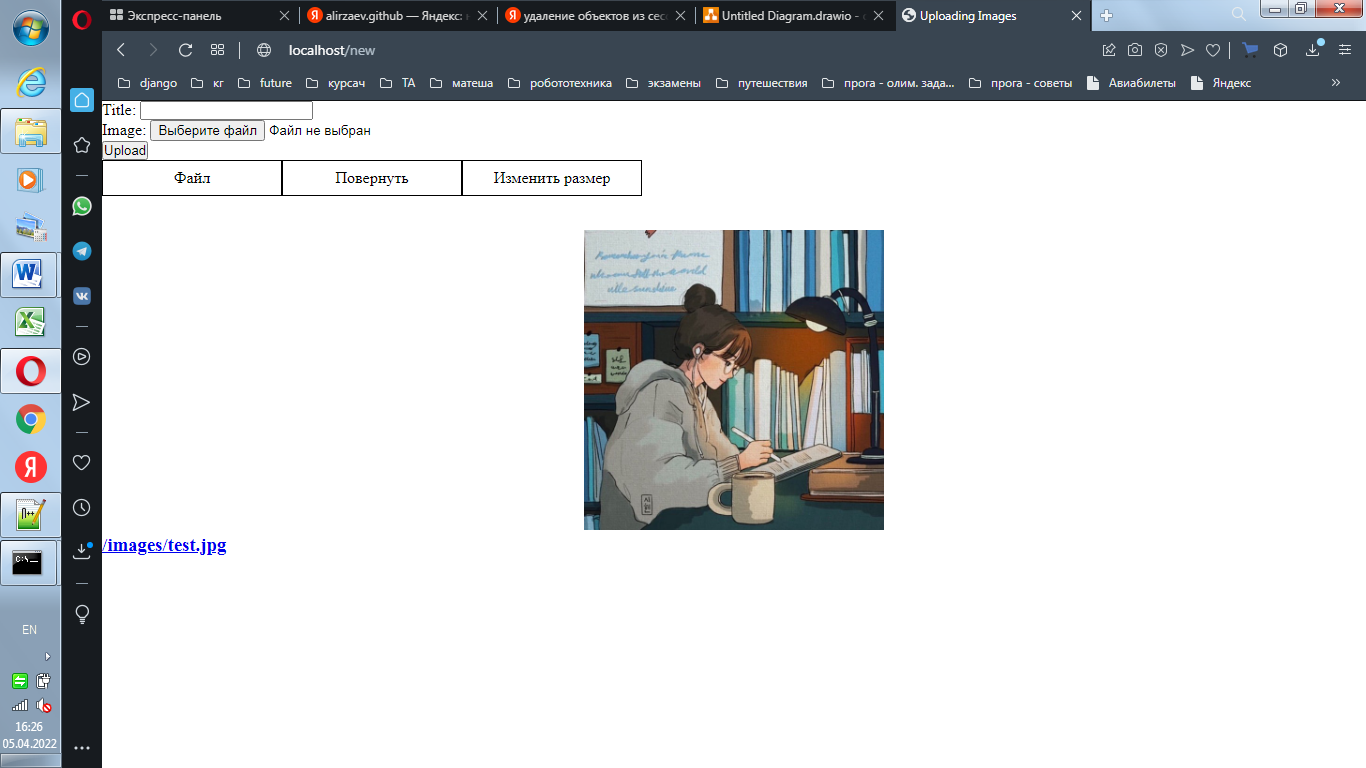


Рисунок 7 – форма для загрузки изображений

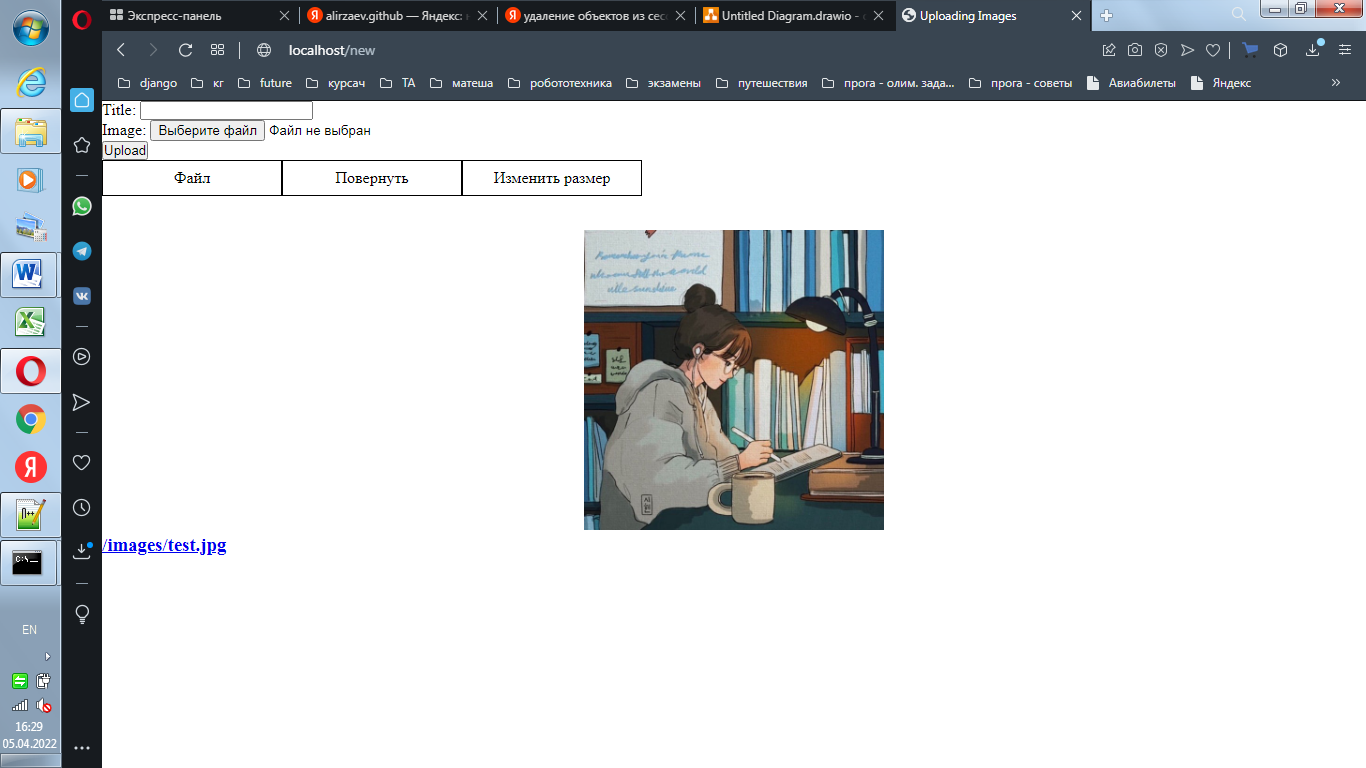


Рисунок 8 – отображение загруженного изображения

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 20 |
| Изм. | Лист | № докум | Подпись | Дата |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 21 |
| Изм. | Лист | № докум | Подпись | Дата |

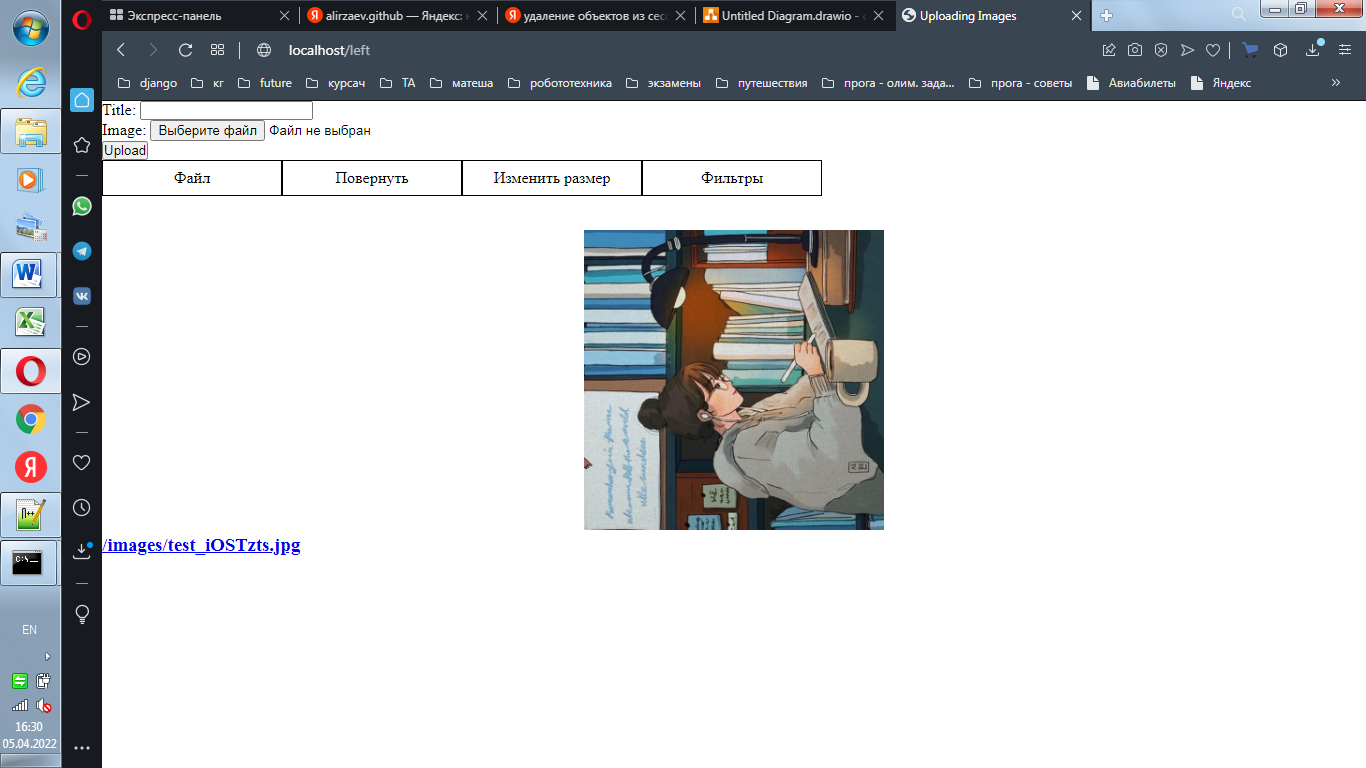


Рисунок 9 – поворот изображения влево

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 22 |
| Изм. | Лист | № докум | Подпись | Дата |

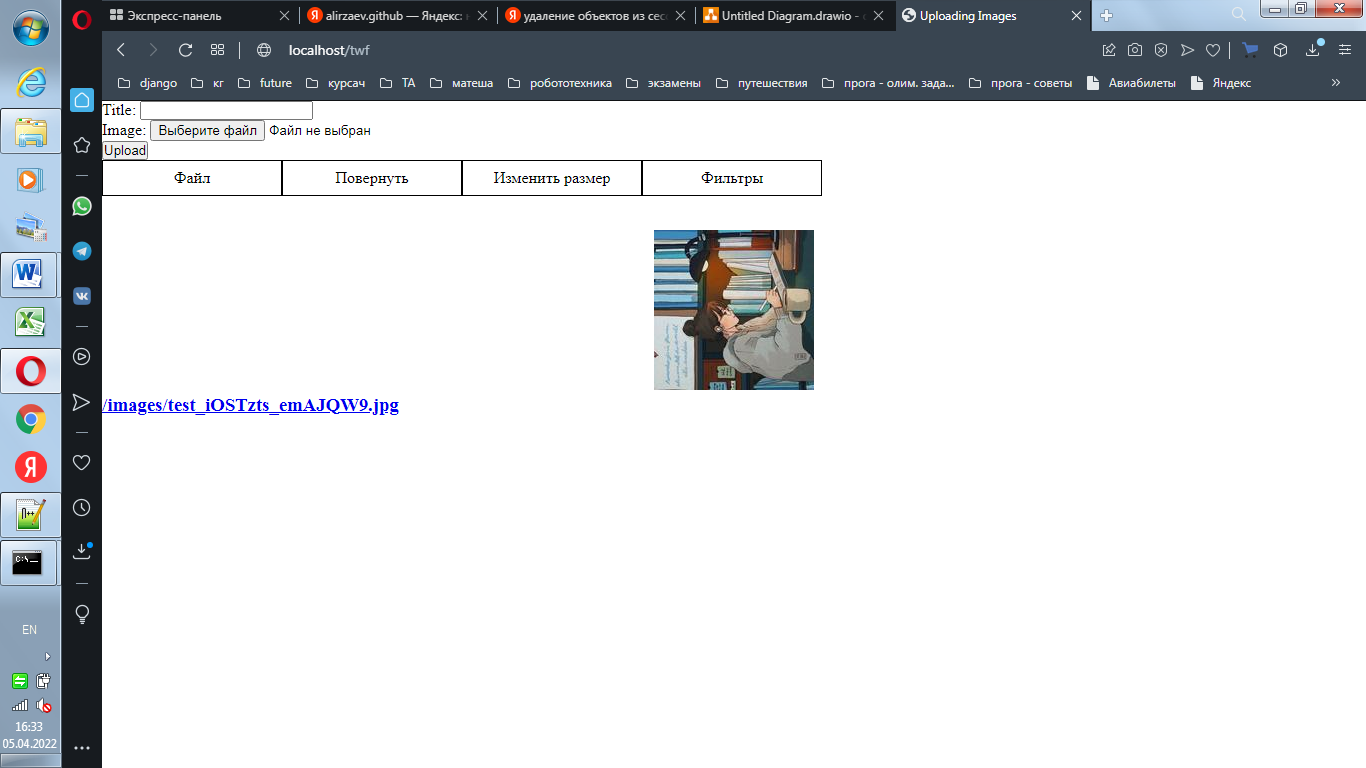


Рисунок 10 – сжатие изображения

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 23 |
| Изм. | Лист | № докум | Подпись | Дата |

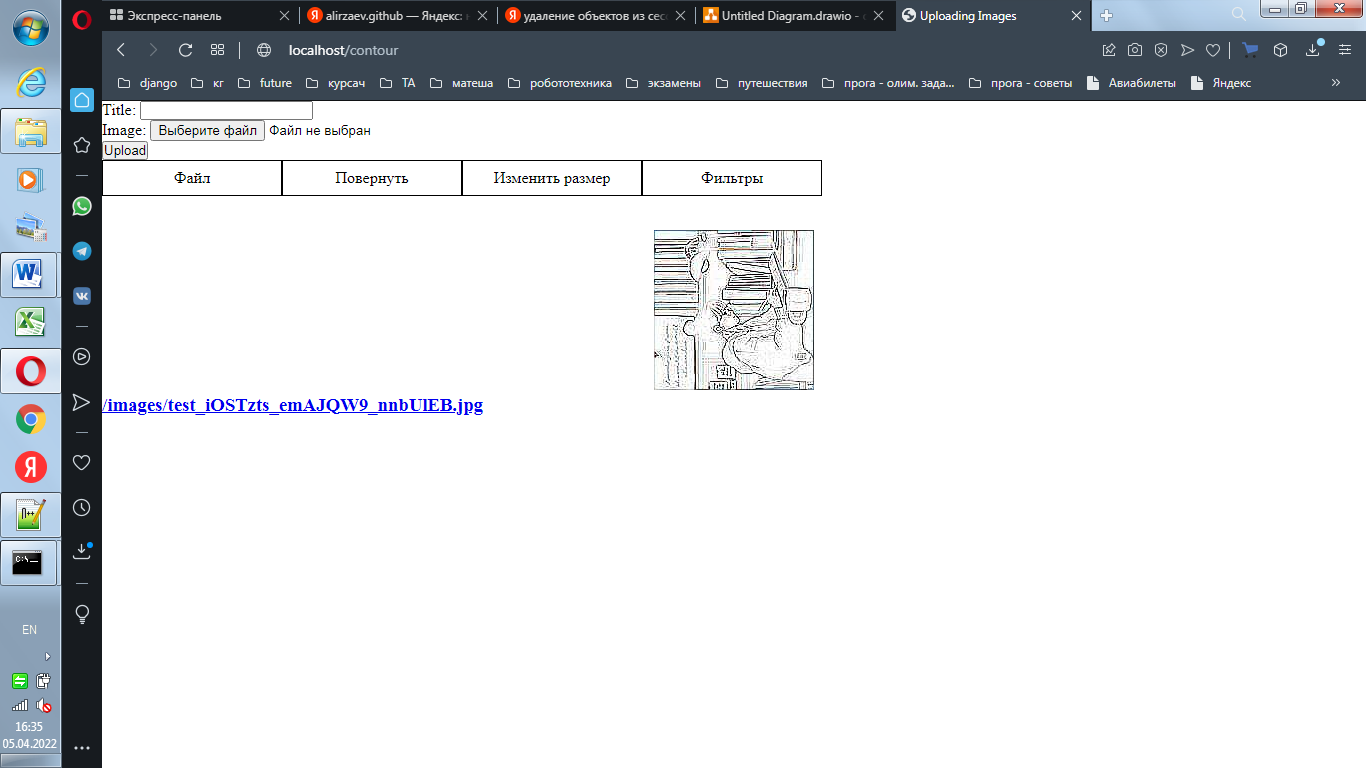


Рисунок 11 – применение фильтра CONTUR

# Приложение В

(обязательное)

Листинг кода

Photo/settings.py

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 24 |
| Изм. | Лист | № докум | Подпись | Дата |

import os.path

from pathlib import Path

# Build paths inside the project like this: BASE\_DIR / 'subdir'.

BASE\_DIR = Path(\_\_file\_\_).resolve().parent.parent

# Quick-start development settings - unsuitable for production

# See https://docs.djangoproject.com/en/3.2/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret!

SECRET\_KEY = 'django-insecure-!enm3c6t^\_o3xgg55am&d)(rmjbsi1+3m$w&x5+-z7usj-z+x@'

# SECURITY WARNING: don't run with debug turned on in production!

DEBUG = True

ALLOWED\_HOSTS = ['localhost', '192.168.1.46']

# Application definition

INSTALLED\_APPS = [

'django.contrib.admin',

'django.contrib.auth',

'django.contrib.contenttypes',

'django.contrib.sessions',

'django.contrib.messages',

'django.contrib.staticfiles',

'loader'

]

MIDDLEWARE = [

'django.middleware.security.SecurityMiddleware',

'django.contrib.sessions.middleware.SessionMiddleware',

'django.middleware.common.CommonMiddleware',

'django.middleware.csrf.CsrfViewMiddleware',

'django.contrib.auth.middleware.AuthenticationMiddleware',

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 25 |
| Изм. | Лист | № докум | Подпись | Дата |

'django.contrib.messages.middleware.MessageMiddleware',

'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

ROOT\_URLCONF = 'photo.urls'

TEMPLATES = [

{

'BACKEND': 'django.template.backends.django.DjangoTemplates',

'DIRS': ['templates'],

'APP\_DIRS': True,

'OPTIONS': {

'context\_processors': [

'django.template.context\_processors.debug',

'django.template.context\_processors.request',

'django.contrib.auth.context\_processors.auth',

'django.contrib.messages.context\_processors.messages',

],

},

},

]

WSGI\_APPLICATION = 'photo.wsgi.application'

# Database

# https://docs.djangoproject.com/en/3.2/ref/settings/#databases

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.sqlite3',

'NAME': BASE\_DIR / 'db.sqlite3',

}

}

# Password validation

# https://docs.djangoproject.com/en/3.2/ref/settings/#auth-password-validators

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 26 |
| Изм. | Лист | № докум | Подпись | Дата |

AUTH\_PASSWORD\_VALIDATORS = [

{

'NAME': 'django.contrib.auth.password\_validation.UserAttributeSimilarityValidator',

},

{

'NAME': 'django.contrib.auth.password\_validation.MinimumLengthValidator',

},

{

'NAME': 'django.contrib.auth.password\_validation.CommonPasswordValidator',

},

{

'NAME': 'django.contrib.auth.password\_validation.NumericPasswordValidator',

},

]

# Internationalization

# https://docs.djangoproject.com/en/3.2/topics/i18n/

LANGUAGE\_CODE = 'en-us'

TIME\_ZONE = 'UTC'

USE\_I18N = True

USE\_L10N = True

USE\_TZ = True

# Static files (CSS, JavaScript, Images)

# https://docs.djangoproject.com/en/3.2/howto/static-files/

STATIC\_URL = '/static/'

#STATIC\_ROOT = 'pythonProject1/photo/static'

STATIC\_ROOT = '/static/'

STATICFILES\_DIRS = [

os.path.join(BASE\_DIR, 'static'),

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 27 |
| Изм. | Лист | № докум | Подпись | Дата |

]

#MEDIA\_URL = '/media/'

MEDIA\_ROOT = os.path.join(BASE\_DIR, 'media/')

# Default primary key field type

# https://docs.djangoproject.com/en/3.2/ref/settings/#default-auto-field

DEFAULT\_AUTO\_FIELD = 'django.db.models.BigAutoField'

#print('Static\_files', STATIC\_URL)

#print('s\_dirs', STATICFILES\_DIRS)

IMAGE\_HOST='http://localhost'

photo/url.py

from django.contrib import admin

from django.urls import path

from django.conf import settings

from django.conf.urls.static import static

from django.urls import path, include

urlpatterns = [

path('admin/', admin.site.urls),

path('', include('loader.urls')),

]

if settings.DEBUG:

#urlpatterns += static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT)

urlpatterns += static(settings.STATIC\_URL, document\_root=settings.STATIC\_ROOT)

Loader/url.py

from django.urls import path

from . import views

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 28 |
| Изм. | Лист | № докум | Подпись | Дата |

urlpatterns = [

path('', views.image\_upload\_view, name='home'),

path('<str:cmd>', views.image\_upload\_view, name='cmds')

]

Loader/views.py

import io

from django.shortcuts import render

from .forms import ImageForm

from django.http import HttpResponse

from .models import Im

from django.conf import settings

import urllib.request

from PIL import Image

from io import BytesIO, StringIO

from django.core.files.base import ContentFile

from . import cmds

def Proverka(url):

if url.split('.')[-1] == 'png':

z = False

else:

z = True

return z

def CheckSession(img\_url, request):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 29 |
| Изм. | Лист | № докум | Подпись | Дата |

if not ('img' in request.session):

request.session['img'] = ''

else:

if img\_url == '':

img\_url = request.session.get('img','')

return img\_url

def ChangeSession(img\_url, request):

if img\_url != "":

if img\_url != request.session.get('img',''):

request.session['img'] = img\_url

else:

if request.session.get('img','') != "":

img\_url = request.session.get('img','')

return img\_url

# Create your views here.

def image\_upload\_view(request, cmd=""):

img\_url = request.GET.get("image", "")

img\_url = CheckSession(img\_url, request)

form = ImageForm()

k = Proverka(img\_url)

if cmd == "":

pass

elif cmd == "new":

img\_url = cmds.new(request)

# k = False

elif cmd == "right":

img\_url = cmds.right(request)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 30 |
| Изм. | Лист | № докум | Подпись | Дата |

elif cmd == "left":

img\_url = cmds.left(request)

elif cmd == "twf":

img\_url = cmds.twf(request)

elif cmd == "fif":

img\_url = cmds.fif(request)

elif cmd == "sev":

img\_url = cmds.sev(request)

elif cmd == "hun":

img\_url = cmds.hun(request)

elif cmd == "hunf":

img\_url = cmds.hunf(request)

elif cmd == "blur":

img\_url = cmds.blur(request)

elif cmd == "sharpen":

img\_url = cmds.sharpen(request)

elif cmd == "contour":

img\_url = cmds.contour(request)

elif cmd == "detail":

img\_url = cmds.detail(request)

elif cmd == "smooth":

img\_url = cmds.smooth(request)

elif cmd == "save":

cmds.save(img\_url, request)

else:

return render(request, '404.html')

img\_url = ChangeSession(img\_url, request)

return render(request, 'index.html', {'form':form, 'img\_url':img\_url, 'k':k} )

Loader/cmds.py

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 31 |
| Изм. | Лист | № докум | Подпись | Дата |

import urllib.request

from .forms import ImageForm

from PIL import Image

from PIL import Image,ImageFilter

from django.conf import settings

from django.core.files.base import ContentFile

from django.core.files import File

from django.utils.text import slugify

from .models import Im

from io import BytesIO

import base64

def new(request):

if request.method == 'POST':

form = ImageForm(request.POST, request.FILES)

if form.is\_valid():

form.instance.session = request.session.session\_key

form.save()

img\_obj = form.instance

img\_url = request.GET.get("image", "")

image\_name = img\_url.split('/')[-1]

image\_format = image\_name.split('.')[-1]

return img\_obj.image.url

return ""

def right(request):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 32 |
| Изм. | Лист | № докум | Подпись | Дата |

try:

img\_url = request.GET.get("image", "")

image\_name = img\_url.split('/')[-1]

image\_format = image\_name.split('.')[-1]

if image\_format == 'jpg':

image\_format = 'JPEG'

#Берем картинку

response = urllib.request.urlopen(settings.IMAGE\_HOST+img\_url)

#Открываем

im = Image.open(ContentFile(response.read()))

im\_rotate = im.rotate(-90)

thumb\_io = BytesIO()

im\_rotate.save(thumb\_io, format=image\_format)

image\_file = ContentFile(thumb\_io.getvalue())

img = Im()

img.session = request.session.session\_key

img.image.save( image\_name, image\_file , save=False)

img.save()

return img.image.url

except:

print("Ошибка при повороте изображения")

return ""

def left(request):

try:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 33 |
| Изм. | Лист | № докум | Подпись | Дата |

img\_url = request.GET.get("image", "")

image\_name = img\_url.split('/')[-1]

image\_format = image\_name.split('.')[-1]

if image\_format == 'jpg':

image\_format = 'JPEG'

#Берем картинку

response = urllib.request.urlopen(settings.IMAGE\_HOST+img\_url)

#Открываем

im = Image.open(ContentFile(response.read()))

im\_rotate = im.rotate(90)

thumb\_io = BytesIO()

im\_rotate.save(thumb\_io, format=image\_format)

image\_file = ContentFile(thumb\_io.getvalue())

img = Im()

img.session = request.session.session\_key

img.image.save( image\_name, image\_file , save=False)

img.save()

return img.image.url

except:

print("Ошибка при повороте изображения")

return ""

def twf(request):

try:

img\_url = request.GET.get("image", "")

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 34 |
| Изм. | Лист | № докум | Подпись | Дата |

image\_name = img\_url.split('/')[-1]

image\_format = image\_name.split('.')[-1]

if image\_format == 'jpg':

image\_format = 'JPEG'

#Берем картинку

response = urllib.request.urlopen(settings.IMAGE\_HOST+img\_url)

#Открываем

im = Image.open(ContentFile(response.read()))

w,h = im.size

w = (w\*25)//100

h = (h\*25)//100

im\_rotate = im.resize((w,h),Image.ANTIALIAS)

thumb\_io = BytesIO()

im\_rotate.save(thumb\_io, format=image\_format)

image\_file = ContentFile(thumb\_io.getvalue())

img = Im()

img.session = request.session.session\_key

img.image.save( image\_name, image\_file , save=False)

img.save()

return img.image.url

except:

print("Ошибка при повороте изображения")

return ""

def fif(request):

try:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 35 |
| Изм. | Лист | № докум | Подпись | Дата |

img\_url = request.GET.get("image", "")

image\_name = img\_url.split('/')[-1]

image\_format = image\_name.split('.')[-1]

if image\_format == 'jpg':

image\_format = 'JPEG'

#Берем картинку

response = urllib.request.urlopen(settings.IMAGE\_HOST+img\_url)

#Открываем

im = Image.open(ContentFile(response.read()))

w,h = im.size

w = (w\*50)//100

h = (h\*50)//100

im\_rotate = im.resize((w,h),Image.ANTIALIAS)

thumb\_io = BytesIO()

im\_rotate.save(thumb\_io, format=image\_format)

image\_file = ContentFile(thumb\_io.getvalue())

img = Im()

img.session = request.session.session\_key

img.image.save( image\_name, image\_file , save=False)

img.save()

return img.image.url

except:

print("Ошибка при повороте изображения")

return ""

def sev(request):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 36 |
| Изм. | Лист | № докум | Подпись | Дата |

try:

img\_url = request.GET.get("image", "")

image\_name = img\_url.split('/')[-1]

image\_format = image\_name.split('.')[-1]

if image\_format == 'jpg':

image\_format = 'JPEG'

#Берем картинку

response = urllib.request.urlopen(settings.IMAGE\_HOST+img\_url)

#Открываем

im = Image.open(ContentFile(response.read()))

w,h = im.size

w = (w\*75)//100

h = (h\*75)//100

im\_rotate = im.resize((w,h),Image.ANTIALIAS)

thumb\_io = BytesIO()

im\_rotate.save(thumb\_io, format=image\_format)

image\_file = ContentFile(thumb\_io.getvalue())

img = Im()

img.session = request.session.session\_key

img.image.save( image\_name, image\_file , save=False)

img.save()

return img.image.url

except:

print("Ошибка при повороте изображения")

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 37 |
| Изм. | Лист | № докум | Подпись | Дата |

return ""

def hun(request):

try:

img\_url = request.GET.get("image", "")

image\_name = img\_url.split('/')[-1]

image\_format = image\_name.split('.')[-1]

if image\_format == 'jpg':

image\_format = 'JPEG'

#Берем картинку

response = urllib.request.urlopen(settings.IMAGE\_HOST+img\_url)

#Открываем

im = Image.open(ContentFile(response.read()))

w,h = im.size

w = (w\*125)//100

h = (h\*125)//100

im\_rotate = im.resize((w,h),Image.ANTIALIAS)

thumb\_io = BytesIO()

im\_rotate.save(thumb\_io, format=image\_format)

image\_file = ContentFile(thumb\_io.getvalue())

img = Im()

img.session = request.session.session\_key

img.image.save( image\_name, image\_file , save=False)

img.save()

return img.image.url

except:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 38 |
| Изм. | Лист | № докум | Подпись | Дата |

print("Ошибка при повороте изображения")

return ""

def hunf(request):

try:

img\_url = request.GET.get("image", "")

image\_name = img\_url.split('/')[-1]

image\_format = image\_name.split('.')[-1]

if image\_format == 'jpg':

image\_format = 'JPEG'

#Берем картинку

response = urllib.request.urlopen(settings.IMAGE\_HOST+img\_url)

#Открываем

im = Image.open(ContentFile(response.read()))

w,h = im.size

w = (w\*150)//100

h = (h\*150)//100

im\_rotate = im.resize((w,h),Image.ANTIALIAS)

thumb\_io = BytesIO()

im\_rotate.save(thumb\_io, format=image\_format)

image\_file = ContentFile(thumb\_io.getvalue())

img = Im()

img.session = request.session.session\_key

img.image.save( image\_name, image\_file , save=False)

img.save()

return img.image.url

except:

print("Ошибка при повороте изображения")

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 39 |
| Изм. | Лист | № докум | Подпись | Дата |

return ""

def blur(request):

try:

img\_url = request.GET.get("image", "")

image\_name = img\_url.split('/')[-1]

image\_format = image\_name.split('.')[-1]

if image\_format == 'jpg':

image\_format = 'JPEG'

#Берем картинку

response = urllib.request.urlopen(settings.IMAGE\_HOST+img\_url)

#Открываем

im = Image.open(ContentFile(response.read()))

im\_rotate = im.filter(ImageFilter.BLUR)

thumb\_io = BytesIO()

im\_rotate.save(thumb\_io, format=image\_format)

image\_file = ContentFile(thumb\_io.getvalue())

img = Im()

img.session = request.session.session\_key

img.image.save( image\_name, image\_file , save=False)

img.save()

return img.image.url

except:

print("Ошибка при повороте изображения")

return ""

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 40 |
| Изм. | Лист | № докум | Подпись | Дата |

def sharpen(request):

try:

img\_url = request.GET.get("image", "")

image\_name = img\_url.split('/')[-1]

image\_format = image\_name.split('.')[-1]

if image\_format == 'jpg':

image\_format = 'JPEG'

#Берем картинку

response = urllib.request.urlopen(settings.IMAGE\_HOST+img\_url)

#Открываем

im = Image.open(ContentFile(response.read()))

im\_rotate = im.filter(ImageFilter.SHARPEN)

thumb\_io = BytesIO()

im\_rotate.save(thumb\_io, format=image\_format)

image\_file = ContentFile(thumb\_io.getvalue())

img = Im()

img.session = request.session.session\_key

img.image.save( image\_name, image\_file , save=False)

img.save()

return img.image.url

except:

print("Ошибка при повороте изображения")

return ""

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 41 |
| Изм. | Лист | № докум | Подпись | Дата |

def contour(request):

try:

img\_url = request.GET.get("image", "")

image\_name = img\_url.split('/')[-1]

image\_format = image\_name.split('.')[-1]

if image\_format == 'jpg':

image\_format = 'JPEG'

#Берем картинку

response = urllib.request.urlopen(settings.IMAGE\_HOST+img\_url)

#Открываем

im = Image.open(ContentFile(response.read()))

im\_rotate = im.filter(ImageFilter.CONTOUR)

thumb\_io = BytesIO()

im\_rotate.save(thumb\_io, format=image\_format)

image\_file = ContentFile(thumb\_io.getvalue())

img = Im()

img.session = request.session.session\_key

img.image.save( image\_name, image\_file , save=False)

img.save()

return img.image.url

except:

print("Ошибка при повороте изображения")

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 42 |
| Изм. | Лист | № докум | Подпись | Дата |

return ""

def detail(request):

try:

img\_url = request.GET.get("image", "")

image\_name = img\_url.split('/')[-1]

image\_format = image\_name.split('.')[-1]

if image\_format == 'jpg':

image\_format = 'JPEG'

#Берем картинку

response = urllib.request.urlopen(settings.IMAGE\_HOST+img\_url)

#Открываем

im = Image.open(ContentFile(response.read()))

im\_rotate = im.filter(ImageFilter.DETAIL)

thumb\_io = BytesIO()

im\_rotate.save(thumb\_io, format=image\_format)

image\_file = ContentFile(thumb\_io.getvalue())

img = Im()

img.session = request.session.session\_key

img.image.save( image\_name, image\_file , save=False)

img.save()

return img.image.url

except:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 43 |
| Изм. | Лист | № докум | Подпись | Дата |

print("Ошибка при повороте изображения")

return ""

def smooth(request):

try:

img\_url = request.GET.get("image", "")

image\_name = img\_url.split('/')[-1]

image\_format = image\_name.split('.')[-1]

if image\_format == 'jpg':

image\_format = 'JPEG'

#Берем картинку

response = urllib.request.urlopen(settings.IMAGE\_HOST+img\_url)

#Открываем

im = Image.open(ContentFile(response.read()))

im\_rotate = im.filter(ImageFilter.SMOOTH)

thumb\_io = BytesIO()

im\_rotate.save(thumb\_io, format=image\_format)

image\_file = ContentFile(thumb\_io.getvalue())

img = Im()

img.session = request.session.session\_key

img.image.save( image\_name, image\_file , save=False)

img.save()

return img.image.url

except:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 44 |
| Изм. | Лист | № докум | Подпись | Дата |

print("Ошибка при повороте изображения")

return ""

def save(img\_url, request):

images = Im.objects.filter(session = request.session.session\_key).exclude(image = img\_url.replace('/', '', 1))

for image in images:

image.image.delete(save = True)

image.delete();

Loader/forms.py

from django import forms

from .models import Im

class ImageForm(forms.ModelForm):

class Meta:

model = Im

fields = ('title','image')

Loader/models.py

from django.db import models

from PIL import Image

from urllib import request

from django.core.files.base import ContentFile

from django.utils.text import slugify

# Create your models here.

class Im(models.Model):

session = models.CharField(max\_length=200)

title = models.CharField(max\_length=200)

image = models.ImageField(upload\_to='images')

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | ТПЖА.09.03.01.487 ПЗ | Лист |
|  |  |  |  |  | 45 |
| Изм. | Лист | № докум | Подпись | Дата |

def \_\_str\_\_(self):

return self.title