Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Базовые компоненты интернет-технологиий»

Лабораторная работа №6.

Выполнил:
студент группы ИУ5-34Б:
Белозеров Дмитрий Сергеевич
Подпись и дата:

Проверил:
преподаватель каф. ИУ5
Гапанюк Юрий Евгеньевич
Подпись и дата:

2022

Задание:

1. Модифицируйте код лабораторной работы №5 или №6 таким образом, чтобы он был пригоден для модульного тестирования.
2. Используя материалы лабораторной работы №4 создайте модульные тесты с применением TDD - фреймворка (2 теста) и BDD - фреймворка (2 теста).

Решение задачи

bot.py

```
import telebot
from telebot import types
import config
import dbworker
import PIL
from PIL import Image
import numpy as np

# Создание бота
bot = telebot.TeleBot(config.TOKEN)


# Начало диалога
@bot.message_handler(commands=['start'])
def cmd_start(message):
    bot.send_message(message.chat.id, 'Я умею соединять 2 картинки в одну и
узнавать их размер')
    dbworker.set(dbworker.make_key(message.chat.id, config.CURRENT_STATE),
config.States.STATE_FIRST_PIC.value)
    bot.send_message(message.chat.id, 'Пришлите первую картинку')


# По команде /reset будем сбрасывать состояния, возвращаясь к началу диалога
@bot.message_handler(commands=['reset'])
def cmd_reset(message):
    bot.send_message(message.chat.id, 'Сбрасываем результаты предыдущего
ввода.')
    dbworker.set(dbworker.make_key(message.chat.id, config.CURRENT_STATE),
config.States.STATE_FIRST_PIC.value)
    bot.send_message(message.chat.id, 'Пришлите первую картинку')


# Обработка первой фотографии
@bot.message_handler(content_types=['photo'], func=lambda message:
dbworker.get(
    dbworker.make_key(message.chat.id, config.CURRENT_STATE)) ==
config.States.STATE_FIRST_PIC.value)
def first_num(message):
    fileID = message.photo[-1].file_id
    file_info = bot.get_file(fileID)
    downloaded_file = bot.download_file(file_info.file_path)
    with open("photos\image_1.jpg", 'wb') as new_file:
        new_file.write(downloaded_file)
    bot.send_message(message.chat.id, f'Вы прислали первую фотографию')
    # Меняем текущее состояние
    dbworker.set(dbworker.make_key(message.chat.id, config.CURRENT_STATE),
config.States.STATE_SECOND_PIC.value)
    dbworker.set(dbworker.make_key(message.chat.id,
config.States.STATE_FIRST_PIC.value), downloaded_file)
```

```
    bot.send_message(message.chat.id, 'Пришлите вторую фотографию')


# Обработка исключений
@bot.message_handler(content_types=['text', 'audio', 'document', 'sticker',
'video', 'video_note', 'voice', 'location',
                                    'contact', 'new_chat_members',
'left_chat_member', 'new_chat_title',
                                    'new_chat_photo',  'delete_chat_photo',
'group_chat_created',
                                    'supergroup_chat_created',
'channel_chat_created', 'migrate_to_chat_id',
                                    'migrate_from_chat_id', 'pinned_message',
'web_app_data'],
                    func=lambda message: dbworker.get(
    dbworker.make_key(message.chat.id, config.CURRENT_STATE)) ==
(config.States.STATE_FIRST_PIC.value))
def error_1(message):
    bot.send_message(message.chat.id, 'Пожалуйста, пришлите картинку!')
    return


@bot.message_handler(content_types=['text', 'audio', 'document', 'sticker',
'video', 'video_note', 'voice', 'location',
                                    'contact', 'new_chat_members',
'left_chat_member', 'new_chat_title',
                                    'new_chat_photo',  'delete_chat_photo',
'group_chat_created',
                                    'supergroup_chat_created',
'channel_chat_created', 'migrate_to_chat_id',
                                    'migrate_from_chat_id', 'pinned_message',
'web_app_data'],
                    func=lambda message: dbworker.get(
    dbworker.make_key(message.chat.id, config.CURRENT_STATE)) ==
(config.States.STATE_SECOND_PIC.value))
def error_2(message):
    # Состояние не изменяется, выводится сообщение об ошибке
    bot.send_message(message.chat.id, 'Пожалуйста, пришлите картинку!')
    return


# Обработка второй фотографии
@bot.message_handler(content_types=['photo'], func=lambda message:
dbworker.get(
    dbworker.make_key(message.chat.id, config.CURRENT_STATE)) ==
config.States.STATE_SECOND_PIC.value)
def second_num(message):
    fileID = message.photo[-1].file_id
    file_info = bot.get_file(fileID)
    downloaded_file = bot.download_file(file_info.file_path)
    with open("photos\image_2.jpg", 'wb') as new_file:
        new_file.write(downloaded_file)
    bot.send_message(message.chat.id, f'Вы прислали вторую фотографию')
    # Меняем текущее состояние
    dbworker.set(dbworker.make_key(message.chat.id, config.CURRENT_STATE),
config.States.STATE_OPERATION.value)
    # Сохраняем первое число
    dbworker.set(dbworker.make_key(message.chat.id,
config.States.STATE_FIRST_PIC.value), downloaded_file)
    markup = types.ReplyKeyboardMarkup(row_width=2)
    itembtn1 = types.KeyboardButton('Узнать размерность каждой картинки')
    itembtn2 = types.KeyboardButton('Соединить 2 картинки в одну')
    markup.add(itembtn1, itembtn2)
    bot.send_message(message.chat.id, 'Нужно сделать тык!!!',
```

```
reply_markup=markup)


#Узнать расширение каждой картинки или соединить картинки
# Выбор действия
@bot.message_handler(func=lambda message: dbworker.get(
    dbworker.make_key(message.chat.id, config.CURRENT_STATE)) ==
config.States.STATE_OPERATION.value)
def operation(message):
    # Текущее действие
    op = message.text
    # Читаем операнды из базы данных
    # Выполняем действие
    list_im = ["photos\image_1.jpg", "photos\image_2.jpg"]
    markup = types.ReplyKeyboardRemove(selective=False)
    if op == 'Узнать размерность каждой картинки':
        result = first_func(list_im)
        bot.send_message(message.chat.id, f'Разрешение первой
фотографии:\n{result[0]} x {result[1]}', reply_markup=markup)
        bot.send_message(message.chat.id, f'Разрешение второй
фотографии:\n{result[2]} x {result[3]}', reply_markup=markup)
    elif op == 'Соединить 2 картинки в одну':
        list_im = ["photos\image_1.jpg", "photos\image_2.jpg"]
        second_func(list_im)
        p = open("photos\Trifecta.jpg", "rb")
        bot.send_photo(message.chat.id, p)
    # Выводим результат
    # Меняем текущее состояние
    dbworker.set(dbworker.make_key(message.chat.id, config.CURRENT_STATE),
config.States.STATE_FIRST_PIC.value)
    # Выводим сообщение
    bot.send_message(message.chat.id, 'Пришлите первую фотографию')



def first_func(list_im):
    img1 = PIL.Image.open(list_im[0])
    img2 = PIL.Image.open(list_im[1])
    wid1, hgt1 = img1.size
    wid2, hgt2 = img2.size
    return [wid1, hgt1, wid2, hgt2]


def second_func(list_im):
    imgs = [PIL.Image.open(i) for i in list_im]
    # выбираем наименьшую картинку и меняем размер 2 фотографии, подстраивая
размер под первую
    min_shape = sorted([(np.sum(i.size), i.size) for i in imgs])[0][1]
    imgs_comb = np.hstack((np.asarray(i.resize(min_shape)) for i in imgs))
    # сохраняем соеденненную картинку
    imgs_comb = PIL.Image.fromarray(imgs_comb)
    imgs_comb.save("photos\Trifecta.jpg")


if __name__ == '__main__':
    bot.infinity_polling()
```

config.py

```
from enum import Enum

# Токент бота
TOKEN = "токен вашего бота"
```

```
# Файл базы данных Vedis
db_file = "db.vdb"

# Ключ записи в БД для текущего состояния
CURRENT_STATE = "CURRENT_STATE"

# Состояния автомата
class States(Enum):
    STATE_START = "STATE_START"  # Начало нового диалога
    STATE_FIRST_PIC = "STATE_FIRST_PIC"
    STATE_SECOND_PIC = "STATE_SECOND_PIC"
    STATE_OPERATION = "STATE_OPERATION"
```

dbworker.py

```
from vedis import Vedis
import config

# Чтение значения
def get(key):
    with Vedis(config.db_file) as db:
        try:
            return db[key].decode()
        except KeyError:
            # в случае ошибки значение по умолчанию - начало диалога
            return config.States.S_START.value


# Запись значения
def set(key, value):
    with Vedis(config.db_file) as db:
        try:
            db[key] = value
            return True
        except:
            # тут желательно как-то обработать ситуацию
            return False


# Создание ключа для записи и чтения
def make_key(chatid, keyid):
    res = str(chatid) + '__' + str(keyid)
    return res
```

Результат

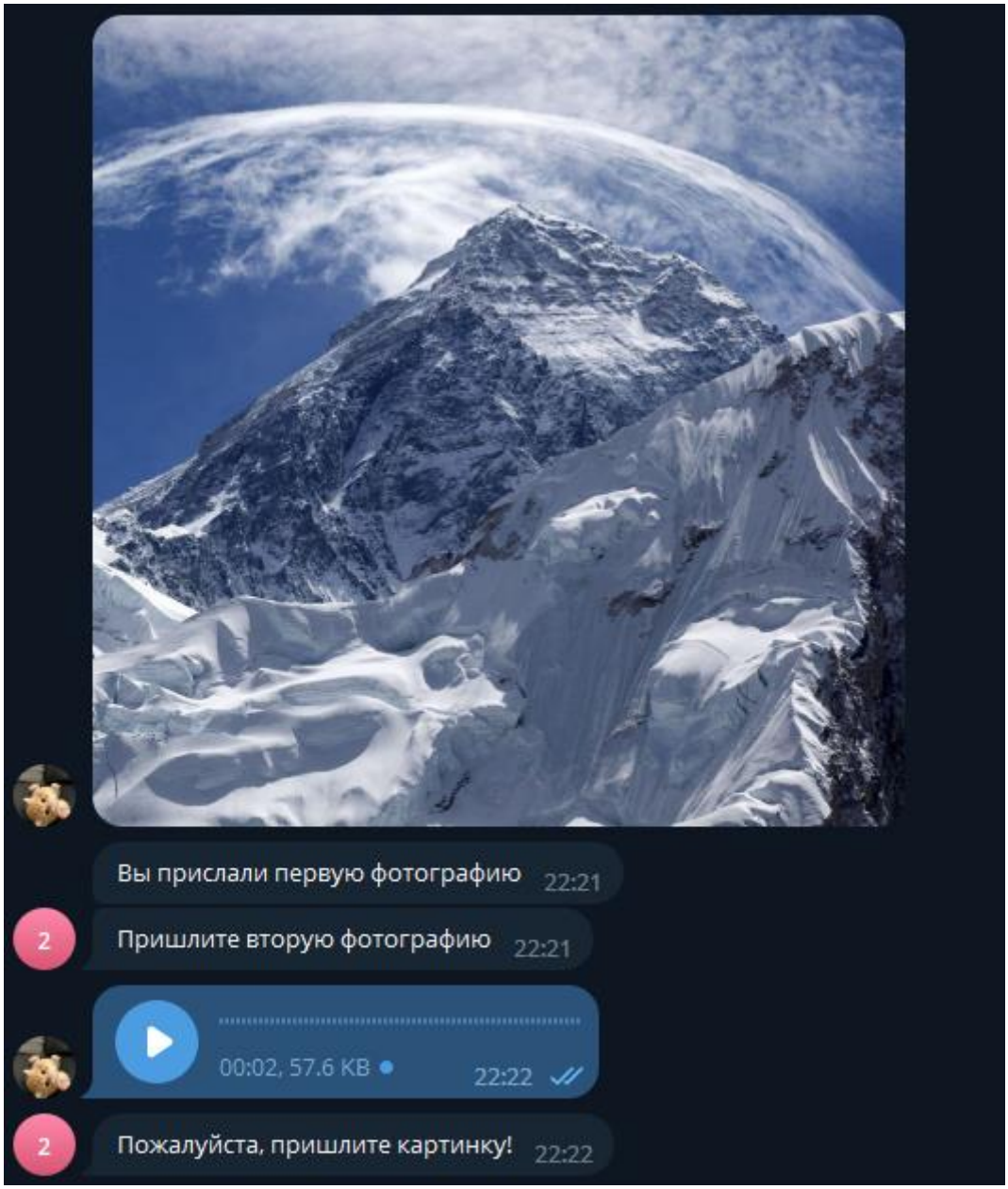

2_in_1_bot
bot
Пожалуйста, пришлите картинку! 19:46
December 12
/reset 22:20
Сбрасываем результаты предыдущего ввода. 22:20
Пришлите первую картинку 22:20
22:20
Вы прислали первую фотографию 22:20
Пришлите вторую фотографию 22:20

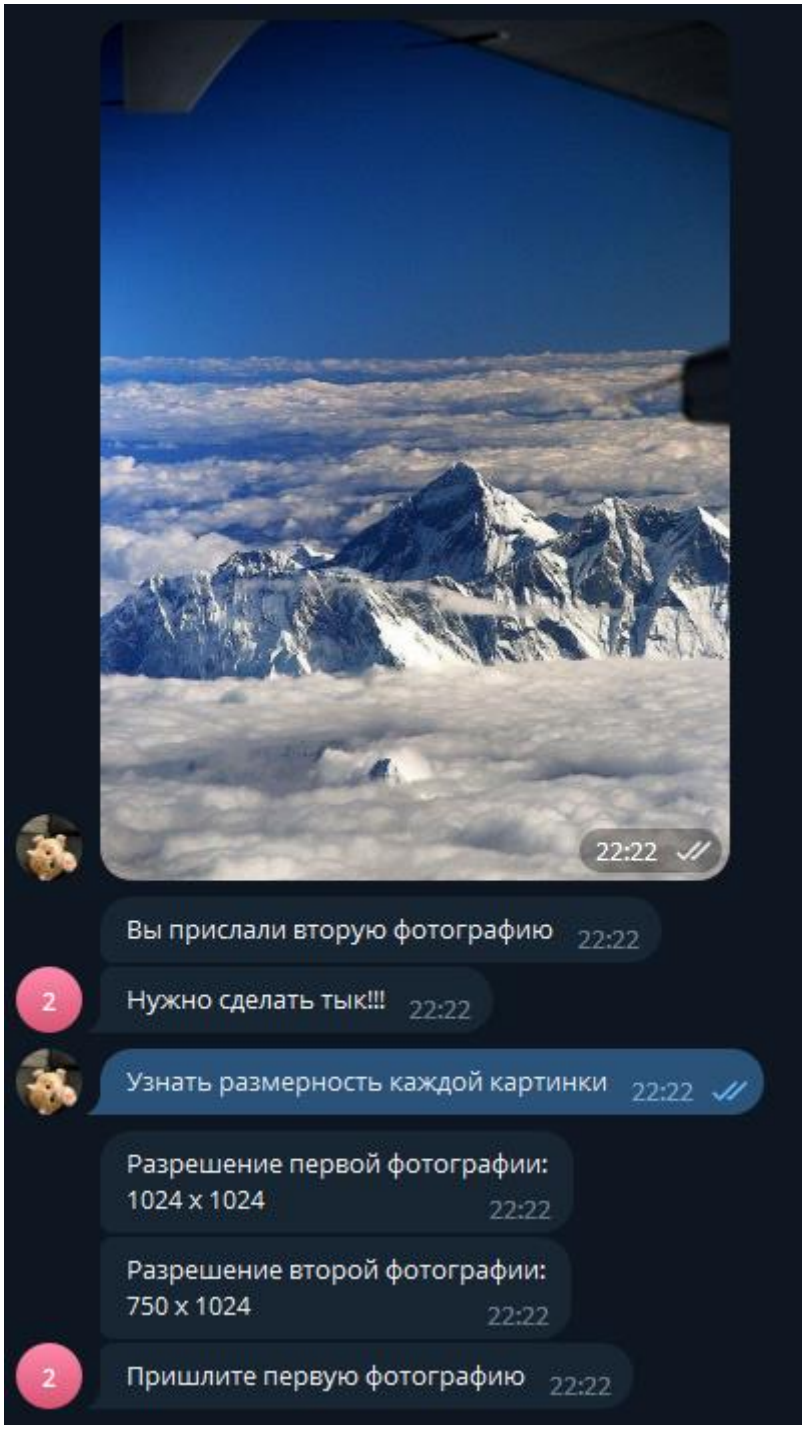
Вы прислали вторую фотографию 22:20

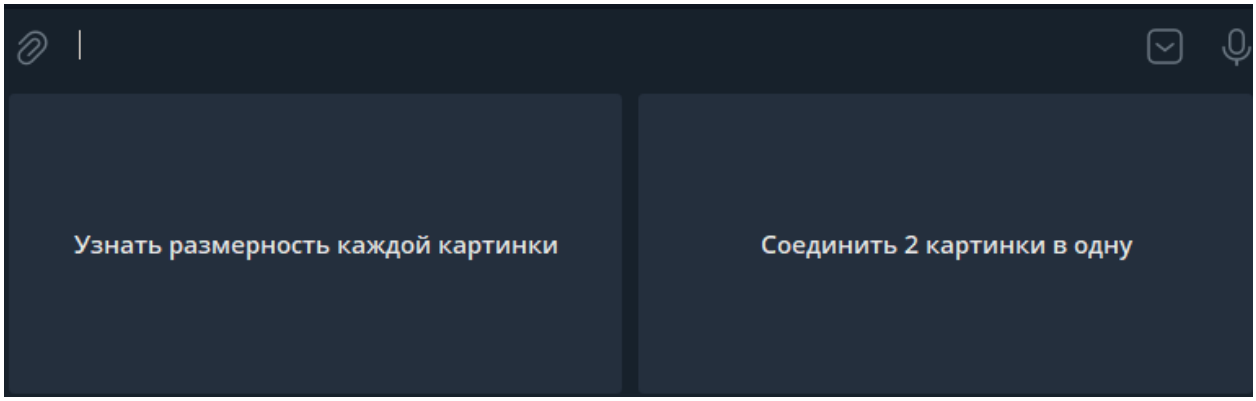
2 Нужно сделать тык!!! 22:20

Соединить 2 картинки в одну 22:20

2 Пришлите первую фотографию 22:20


Вы прислали первую фотографию 22:21
2
Пришлите вторую фотографию 22:21
00:02, 57.6 KB
22:22
2
Пожалуйста, пришлите картинку! 22:22


22:22
Вы прислали вторую фотографию 22:22
2
Нужно сделать тык!!! 22:22
Узнать размерность каждой картинки 22:22
Разрешение первой фотографии:
1024 x 1024 22:22
Разрешение второй фотографии:
750 x 1024 22:22
2
Пришлите первую фотографию 22:22


Узнать размерность каждой картинки
Соединить 2 картинки в одну