Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**Иркутский национальный исследовательский   
технический университет**

|  |
| --- |
| Институт информационных технологий и анализа данных |
| наименование института |

|  |
| --- |
| Отчет по дисциплине |
| «Управление разработкой программного обеспечения» |
| по теме: |
| **«Создание чат-бота»** |
|  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил студент группы |  | АСУб-21-1 |  |  |  | С.Н. Коломак |
|  |  | Шифр группы |  | Подпись |  | И.О. Фамилия |
| Проверил преподаватель |  |  |  |  |  | Т.А. Моргунова |
|  |  |  |  | Подпись |  | И.О. Фамилия |

Иркутск 2024 г.

Содержание

[1 Постановка цели и задач 3](#_Toc162299281)

[2 Моделирование замкнутой системы 4](#_Toc162299282)

[3 Вывод выражений, для функций чувствительности 6](#_Toc162299283)

[4 Алгоритм метода наискорейшего спуска 8](#_Toc162299284)

[5 Картины сходимости метода 9](#_Toc162299285)

[6 Зависимости , , 11](#_Toc162299286)

[7 Зависимости 14](#_Toc162299287)

[8 Графики переходных процессов 16](#_Toc162299288)

[9 Листинг 17](#_Toc162299289)

# 1 Бриф заказчика

Проблематика:

Менеджер проектов не успевает отслеживать сроки действия хостинга у клиентов, вследствие чего хостинг отключают за неуплату. Для планового и оперативного уведомления менеджера о сроках оплаты принято решение разработать Telegram-бота.

Что необходимо сделать?

Разработать чат-бот в Telegram, который позволяет по базе номеров в таблице находить пользователей и добавлять их в телеграм-канал, а также присылать сообщения о сроках уплаты хостинга.

1. Получать номера телефонов из табличного документа и сохранять id пользователей, соответствующих номерам в документе. Добавлять контакты по id. Лимит добавлений не должен превышать 200 контактов в сутки.

Необходимо предусмотреть возможность работы бота с локальными файлами на компьютере пользователя, а также с файлами, доступными по ссылке через сервисы Google Docs и Яндекс.Документы.

2. При разработке, используя API хостинга Timeweb, к примеру, необходимо предусмотреть, что чат-бот должен присылать уведомление о том, что подходит срок оплаты хостинга. Ваш чат-бот должен присылать первое уведомление за 1 месяц до окончания срока действия хостинга. Если с момента первого уведомления прошло больше 2 недель и оплата не поступила, бот должен присылать уведомления раз в два дня до момента поступления денег на счет аккаунта.

Текст сообщения должен быть следующий:

«Оплата хостинга [номер аккаунта] на сумму [сумма]. Хостинг отключится [дата]»

Функциональность бота:

⁃ уведомлять о сроках оплаты хостинга

⁃ добавлять новые аккаунты для мониторинга

⁃ отключать мониторинг сроков оплаты хостинга у определенных аккаунтов

⁃ проверять функциональность бота (внутреннее тестирование и контроль работоспособности).

# 2 Задание и описание функциональности

Требования к проекту:

Перед началом работы подготовить блок-схему работы чат-бота, отобразить зависимости, показать логику работы. Обосновать выбор стека разработки. Объяснить причину его выбора, сравнить с другими возможными реализациями. Обосновать эффективность выбранного стека.

Разработку проекта и документирования вести в Githab. Обратить внимание на качество документации. Документация к коду должна быть обязательно.

Разберём некоторые особенности брифа заказчика:

Так как бота в приложении Telegram не могут самостоятельно начать диалог с пользователем без его согласия (т.е. нужно чтобы пользователь самостоятельно в первый раз начал диалог с ботом), то и добавление именно телеграмм ботом в группу пользователей невозможно. Поэтому этот пункт из брифа отпадает, из-за невозможности его реализовать. Также отсутствует необходимость и добавлять все аспекты для работы с группой.

Проанализировав ситуацию, можно сделать следующий анализ о функциональности чат-бота:

* уведомлять о сроках оплаты хостинга автоматически, а также по желанию менеджера;
* отключать пользователей по истечению срока оплаты;
* предоставлять менеджеру краткую информацию о пользователях, у которых близится срок оплаты.

Подключать пользователей к боту должен сам менеджер. А от концепции наличии группы тогда можно полностью отказаться.

# 3 Выбор и обоснование технологий для разработки

Создание чат бота будет осуществлено с использованием следующих технологий:

* Язык программирования: Python. Это высокоуровневый интерпретируемый язык программирования с динамической типизацией. Имеет неописуемое наличие библиотек, предназначенных для различных задач.
* Библиотеки:
  + telebot – библиотека, предназначенная для работы с API Telegram, которая предоставляет простой и удобный интерфейс для создания и управления чат-ботами. Функциональность: позволяет отправлять сообщения, обрабатывать входящие сообщения, настраивать клавиатуры и многое другое.Преимущества: Легкость использования, обширная документация, активное сообщество, поддержка различных типов сообщений и клавиатур.
  + gspread – это библиотека Python, специально разработанная для удобной работы с Google Sheets через их API. Она обеспечивает простой и интуитивно понятный интерфейс для создания, чтения, обновления и удаления данных в таблицах Google Sheets. Преимущества: простота использования, обширная документация, активное сообщество, поддержка различных типов данных.
  + schedule – это библиотека Python, предназначенная для управления выполнением задач по расписанию. Она позволяет запускать функции или вызывать события в определенное время или с интервалом. Преимущества: простота использования, интуитивный интерфейс, удобная настройка расписания, поддержка различных форматов времени. Она неплохо работает вместе со следующими библиотеками:
    - time – это стандартная библиотека Python, которая предоставляет функции для работы с временем. Она позволяет измерять прошедшее время, управлять задержками и таймерами, а также работать с датами и временем в различных форматах.
    - dateutil.relativedelta – это часть библиотеки dateutil в Python, которая предоставляет функциональность для выполнения математических операций с датами. Она позволяет выполнять относительные операции с датами, такие как добавление или вычитание определенного количества времени или временных интервалов к заданной дате.
  + oauth2client.service\_account – это модуль библиотеки OAuth2Client в Python, который предоставляет функциональность для аутентификации при использовании сервисного аккаунта Google. Сервисные аккаунты используются для автоматизированного доступа к ресурсам Google, таким как Google Sheets, Google Drive и другим API. Модуль позволяет загружать учетные данные сервисного аккаунта из JSON-файла и использовать их для установления безопасного соединения с API Google.

Наличие вышеописанных технологий, позволит без проблем создать телеграмм бота, с возможностью подключения к Google Sheets, а также манипуляцией с временными данными.

# 4 Создания файла сохранения информации

Опишем структуру таблицы данных, которая необходимо для работоспособности приложения:

id\_user – уникальное имя клиента (формат данных – текстовый);

id\_telegram – уникальный номер клиента в Telegram (формат данных – цифровая последовательность);

active/noactive – состояние клиента. Может быть active или noactive;

data\_last\_pay – дата последнего платежа;

payment\_amount – сумма платежа(формат данных – цифровая последовательность);

phone\_number – номер телефона клиента(формат данных – цифровая последовательность);

Таблица – Таблица данных

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| id\_user | id\_telegram | active/noactive | data\_last\_pay | payment\_amount | phone\_number |
| sergey | 475776058 | active | 02.03.2024 | 5000 | 79500732887 |
| bulat | 806304402 | active | 06.03.2024 | 6000 | 79999999999 |
| … | … | … | … | … | … |

# 5 Схема взаимодействия пользователя с чат-ботом и обмена данными с сервером

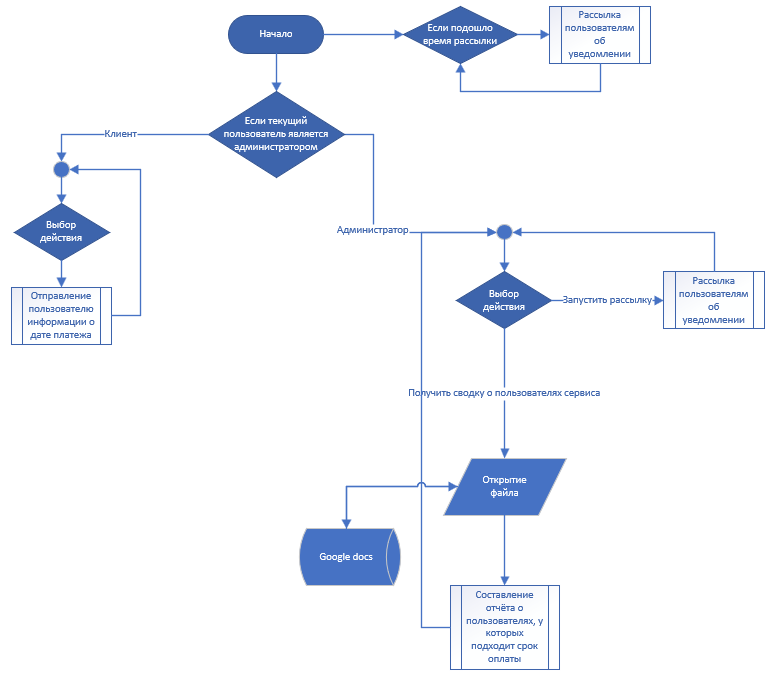


Рисунок – Общая схема взаимодействия с приложением

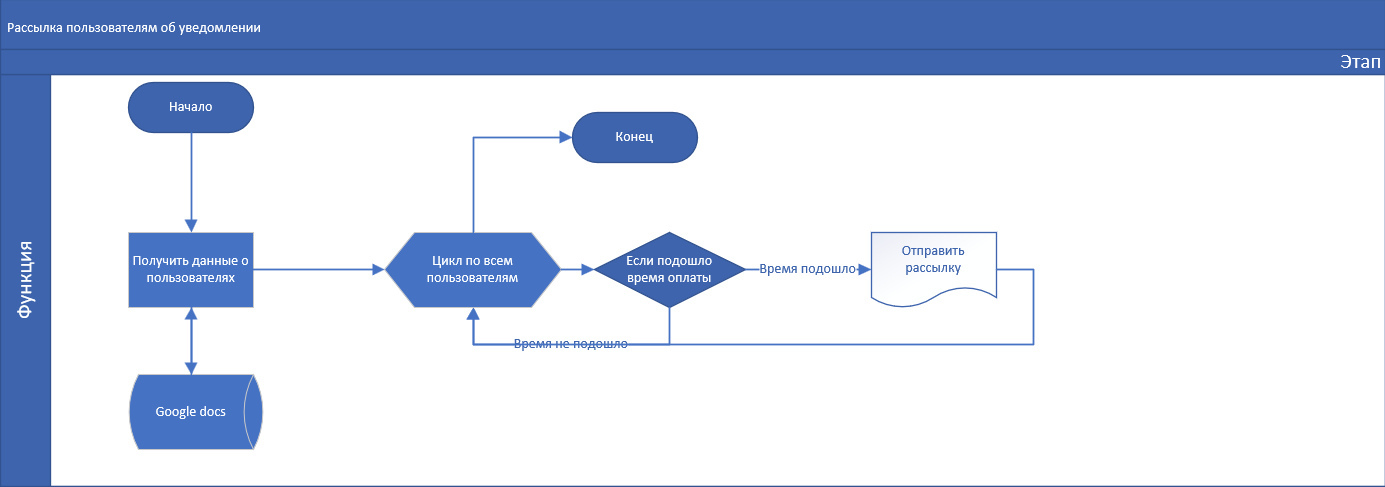


Рисунок – Блок-схема с рассылкой информации пользователям

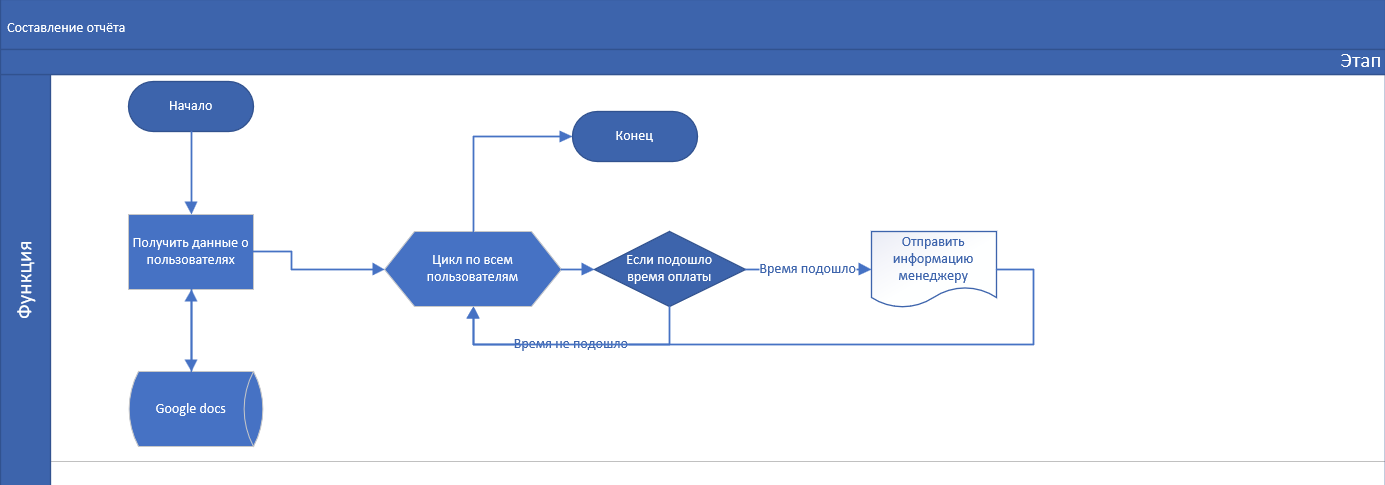


Рисунок – Блок-схема с составлением отчёта для менеджера

# 6 Написание технического задания

1. Общие сведения
   1. Наименование проекта: Telegram-бот для управление и менеджментом информации о сроках оплаты клиентами сервиса.
   2. Заказчик: Моргунова Т.А.
   3. Исполнитель: Коломак С.Н.
   4. Сроки реализации: 28.03.2024
2. Цели и назначения создания автоматизированной системы
   1. Создание удобного инструмента для управления информацией о клиентах и оплатах через мессенджер Telegram.
   2. Предоставление возможности администраторам и пользователям быстро получать информацию о статусе оплаты и ближайших сроках оплаты.
3. Характеристика объектов автоматизации
   1. Система должна обрабатывать информацию о клиентах, их контактных данных и сроках оплаты.
4. Требования к автоматизированной системе
   1. Система должна быть интегрирована с мессенджером Telegram.
   2. Должны быть предусмотрены различные уровни доступа: администратор и пользователь.
   3. Система должна отправлять уведомления о ближайших сроках оплаты.
   4. Возможность запроса информации о ближайшей дате оплаты для конкретного пользователя.
   5. Возможность администратору получать информацию о клиентах с определенными характеристиками.
   6. Возможность администратору отправить уведомления о сроках оплаты пользователям.
5. Состав и содержание работ по созданию автоматизированной системы
   1. Разработка бота для Telegram.
   2. Интеграция с базой данных клиентов.
   3. Разработка логики уведомлений о сроках оплаты.
   4. Создание интерфейса для администратора и пользователя.
6. Порядок разработки автоматизированной системы
   1. Анализ требований и составление технического задания.
   2. Разработка структуры базы данных.
   3. Создание бота для Telegram.
   4. Разработка логики работы с данными.
   5. Тестирование и отладка системы.
   6. Внедрение и обучение пользователей с использованием «инструкций использования» (см. пункт 7.1).
7. Порядок контроля и приёмки автоматизированной системы
   1. Контроль качества разработки на каждом этапе.
   2. Тестирование функциональности и стабильности системы.
   3. Проведение приёмочных испытаний с участием заказчика.
8. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу автоматизированной системы в действие
   1. Подготовка базы данных с информацией о клиентах и оплатах.
   2. Обеспечение стабильного интернет-соединения для работы бота.

# 7 Документация

## 7.1 Инструкция использования

Начало:

1. Пользователь запускает бота, отправив команду /start или начинает взаимодействие с ботом.
2. Идентификация пользователя:
   1. Бот проверяет роль пользователя: если пользователь явялется администратором, то показывает ему меню взаимодействия администратора, иначе бот показывает меню взаимодействия пользователя.

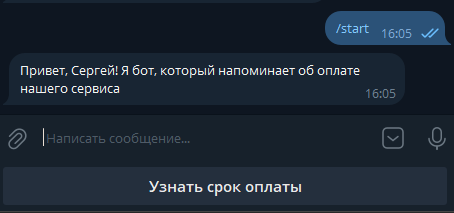


Рисунок – Начало диалога с чат-ботом для клиента

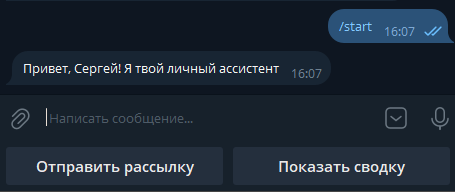


Рисунок – Начало диалога с чат-ботом для менеджера

Выбор действия:

1. Если пользователь – администратор: Бот предоставляет администратору возможность выбрать действие из предложенных вариантов:
   1. отправить рассылку пользователям:



Рисунок – Кнопка "Отправить рассылку"

* 1. показать сводку о пользователях сервиса:



Рисунок – Кнопка "Показать сводку"

1. Если пользователь – обычный пользователь: Бот предоставляет пользователю возможность узнать информацию о ближайшей дате оплаты.



Рисунок – Кнопка "Узнать срок оплаты"

Обработка действий:

1. В зависимости от выбранного действия:
2. Бот выполняет соответствующую операцию (например, отправляет список пользователей с определенными сроками оплаты администратору или информацию о ближайшей дате оплаты пользователю).

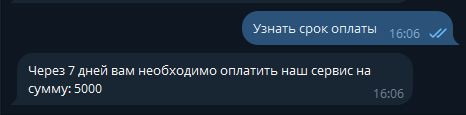


Рисунок – Результат действия "Узнать срок оплаты"

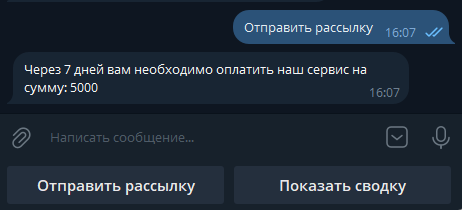


Рисунок – Результат действия "Отправить рассылку"

Данный результат может смутить, но бот работает корректно, просто для отладки сам менеджер был внесён в общий список чтобы проверить работоспособность системы.

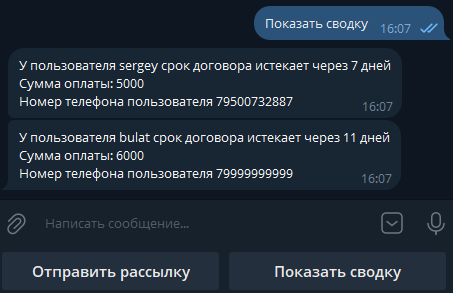


Рисунок – Результат действия "Показать сводку"

Завершение диалога:

1. После выполнения операции бот предоставляет пользователю возможность продолжить взаимодействие с ботом (например, отправив еще одну команду) или завершить диалог.

## 7.2 Инструкция эксплуатации

Запуск чат-бота:

1. Для начала работы чат-бота необходимо запустить приложение run.py. При этом должны быть соблюдены следующие особенности:
   1. Рядом с файлом run.py в одной директории (папке) должен находится файл «service\_account.json», необходимый приложению для работы с Google Sheets.
   2. У вас должен быть установлен Python версии 3.7 и выше. Также необходимо чтобы были установлены библиотеки: telebot; schedule; gspread; oauth2client; datetime.
      1. Если у вас не установлены следующие библиотеки, то воспользуйтесь командой pip install «название\_библиотеки» в командной строке для её установки.
2. При возникновении неисправностей приложения, перезапустите файл run.py

Описание функций приложения:

1. Функции для отрисовки меню:
   1. draw\_manager\_menu – создаёт меню взаимодействия для менеджера. Входных параметров нет. Возвращает готовое меню.
   2. draw\_user\_menu – создаёт меню взаимодействия для клиента. Входных параметров нет. Возвращает готовое меню.
2. Функции обработки сообщений пользователя:
   1. start(message) – начало диалога с пользователем. Входной параметр: message – сообщение от пользователя. Если сообщением пользователя была команда начала диалога («/start»), то программа отправит приветственное сообщение и отрисует меню взаимодействия. Возвращаемых параметров нет.
   2. mailing(message) – обработчик текстовых сообщений пользователя (нажатия на кнопки меню). Входной параметр: message – сообщение от пользователя. Производит определённые действия в соответствии с написанным пользователем сообщением. Возвращаемых параметров нет.
   3. SendMsg(send=False) – обработчик отправления рассылки. Опциональный входной параметр: send – флаг, который говорит о том, что необходимо провести рассылку всем пользователям (по умолчанию равен False). Отправляет уведомление об оплате пользователям. Возвращаемых параметров нет.
   4. GetInfo() – функция для составления отчёта менеджеру. Входных параметров нет. Составляет отчёт по пользователям системы и отправляет его менеджеру. Возвращаемых параметров нет.
   5. GetMsg(user\_id) – функция отправки информации об оплате определённому пользователю. Входной параметр: user\_id – идентификатор пользователя, которому необходимо отправить сообщение. Получает информацию об указанном пользователе и отправляет ему рассылку. Выходных параметров нет.
3. Вспомогательные функции:
   1. days\_between(d1,d2) – функция для подсчёта разницы между днями. Входные параметры: d1 – первый день, d2 – второй день. Возвращает разницу между днями.
   2. day\_token(day\_to) – функция для определения склонения слова «день». Входной параметр: day\_to – номер дня, для которого необходимо просклонять слово «день». Возвращает правильное склонение слова «день».
   3. start\_polling() – функция запускающая Telegram-бота во втором потоке. Входных параметров нет. Выходных параметров нет.

# 9 Листинг

# -\*- coding: utf-8 -\*-

"""lab1.ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/1EbmPdkib8SZYYnv7gBLtOlPJ-G4nKHXV

"""

#pip install --upgrade gspread

#pip install schedule

#pip install telebot

import telebot

from datetime import date

from datetime import datetime

from telebot import types

import time

import threading

import schedule

import gspread

from dateutil.relativedelta import relativedelta

token="6604001463:AAFxz5ubGfdLRQUDwI4MigVWQ2w-MFfLyDA"

bot=telebot.TeleBot(token)

#sa = gspread.service\_account(filename="service\_account.json")

from gspread import authorize

from oauth2client.service\_account import ServiceAccountCredentials

scopes = ["https://spreadsheets.google.com/feeds",

"https://www.googleapis.com/auth/spreadsheets",

"https://www.googleapis.com/auth/drive",

"https://www.googleapis.com/auth/drive"]

cred = ServiceAccountCredentials.from\_json\_keyfile\_name("service\_account.json", scopes)

sa = authorize(cred)

sh = sa.open("yrpolab1")

wks = sh.worksheet("user\_info")

headers = wks.row\_values(1)

def days\_between(d1, d2):

return (d1 - d2).days

def draw\_manager\_menu():

markup = types.ReplyKeyboardMarkup(resize\_keyboard=True)

btn1 = types.KeyboardButton("Отправить рассылку")

btn2 = types.KeyboardButton("Показать сводку")

markup.add(btn1, btn2)

return markup

def draw\_user\_menu():

markup = types.ReplyKeyboardMarkup(resize\_keyboard=True)

btn1 = types.KeyboardButton("Узнать срок оплаты")

markup.add(btn1)

return markup

@bot.message\_handler(commands=['start'])

def start(message):

#475776058

if message.from\_user.id == (475776058):

markup=draw\_manager\_menu()

bot.send\_message(message.chat.id, text="Привет, {0.first\_name}! Я твой личный ассистент".format(message.from\_user), reply\_markup=markup)

else:

markup=draw\_user\_menu()

bot.send\_message(message.chat.id, text="Привет, {0.first\_name}! Я бот, который напоминает об оплате нашего сервиса".format(message.from\_user), reply\_markup=markup)

day\_token = lambda day\_to: (

(day\_to in range(5, 20)) and 'дней' or

(1 in (day\_to, (diglast := day\_to % 10))) and 'день' or

({day\_to, diglast} & {2, 3, 4}) and 'дня' or 'дней')

def SendMsg(send=False):

dataBase = wks.get\_all\_records()

for user in dataBase:

dataOfUser=str(user['data\_last\_pay'])

dataOfUser=datetime.strptime(dataOfUser, '%d.%m.%Y') + relativedelta(months=+1)

today=datetime.strptime(str(date.today()), '%Y-%m-%d')

betwens=days\_between(dataOfUser,today)

payment\_sum=user['payment\_amount']

if ((betwens == 30 or (betwens<=15 and betwens>0 and betwens%2==1)) or send):

if (user['active/noactive'] == 'active'): bot.send\_message(user['id\_telegram'],f'Через {betwens} {day\_token(betwens)} вам необходимо оплатить наш сервис на сумму: {payment\_sum}')

else: continue

elif betwens>30:

bot.send\_message(user['id\_telegram'],'В скором времени вас отключат от хостинга')

rowToUpdate = wks.find(str(user['id\_telegram'])).row

colToUpdate = headers.index('active/noactive')+1

cellToUpdate = wks.cell(rowToUpdate, colToUpdate)

cellToUpdate.value = 'noactive'

wks.update\_cells([cellToUpdate])

def GetInfo(admin):

dataBase = wks.get\_all\_records()

dataBase.sort(key=lambda k : k['data\_last\_pay'])

for user in dataBase:

dataOfUser=str(user['data\_last\_pay'])

dataOfUser=datetime.strptime(dataOfUser, '%d.%m.%Y') + relativedelta(months=+1)

today=datetime.strptime(str(date.today()), '%Y-%m-%d')

betwens=days\_between(dataOfUser,today)

user\_name=user['id\_user']

payment\_sum=user['payment\_amount']

phone=user['phone\_number']

if betwens<=15 and betwens>0:

bot.send\_message(admin,f'У пользователя {user\_name} срок договора истекает через {betwens} {day\_token(betwens)}\nСумма оплаты: {payment\_sum}\nНомер телефона пользователя {phone}')

def GetMsg(user\_id):

user=wks.row\_values(wks.find(str(user\_id)).row)

dataOfUser=str(user[3])

dataOfUser=datetime.strptime(dataOfUser, '%d.%m.%Y') + relativedelta(months=+1)

today=datetime.strptime(str(date.today()), '%Y-%m-%d')

betwens=days\_between(dataOfUser,today)

payment\_sum=user[4]

bot.send\_message(user\_id,f'Через {betwens} {day\_token(betwens)} вам необходимо оплатить наш сервис на сумму: {payment\_sum}')

@bot.message\_handler(content\_types=['text'])

def mailing(message):

if (message.text == "Отправить рассылку"):

SendMsg(send=True)

elif (message.text == "Узнать срок оплаты"):

GetMsg(message.from\_user.id)

elif (message.text == "Показать сводку"):

GetInfo(message.from\_user.id)

else:

bot.send\_message(message.from\_user.id,'Неизвестная комманда, попробуйте ещё раз.')

#bot.polling(none\_stop=True)

def start\_polling():

bot.infinity\_polling(none\_stop=True)

polling\_thread = threading.Thread(target=start\_polling)

polling\_thread.start()

schedule.every().day.at("23:54").do(SendMsg)

try:

while True:

schedule.run\_pending()

time.sleep(1)

except:

pass