Пояснительная записка к дипломной работе «Telegram бот»

студента Салмина Р.А.

1. Аннотация

В рамках дипломной работы на языке Python был реализован проект «Telegram бот» (бот). Бот имеет следующие интерфейсы:

- пользователя;

- администратора.

Интерфейс пользователя позволяет просматривать сообщения бота, вводить необходимые параметры, оплачивать услуги и общаться с администратором бота.

Интерфейс администратора: просмотр всех логов бота и пользователей, отвечать на вопросы пользователей.

1. Техническое задание

Представлен шаблон в формате «word» в котором представлены формулы для просчета значений. Формулы прописаны по типу: переменная 1 \* на переменную 2 и т.д. (просто текстом).

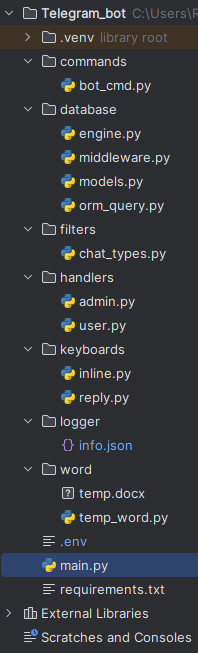
Необходимо сделать «Telegram бот», со следующим функционалом:

* запросить пользователя значения параметров;
* преобразовывать значения по формулам;
* заполнить шаблон на основании этих значений;
* запросить оплату у пользователя (интеграция с «юкассой») с проверкой платежа;
* отправить пользователю файл формате «word».

1. Реализация

Перед написанием кода в программном обеспечении «Telegram», с помощью «BotFather», зарегистрирован бот проекта с необходимыми настройками для работы.

Согласно выбранному курсу обучения, проект был реализован на языке «Python 3.10» с использованием среды разработки «PyCharm 2023.3.2».

Для удобства работы было создано виртуальное окружение в которое устанавливались все необходимые библиотеки. Весь перечень применяемых библиотек представлен в файле «requirements.txt» и дублируется в приложение 1. Определены глобальные константы в файле «.env» с использованием библиотеки «python-dotenv1.0.1».

Функционал бота распределен на соответствующие модули (папки, файлы).

За основу работы с ботом были выбраны библиотеки:

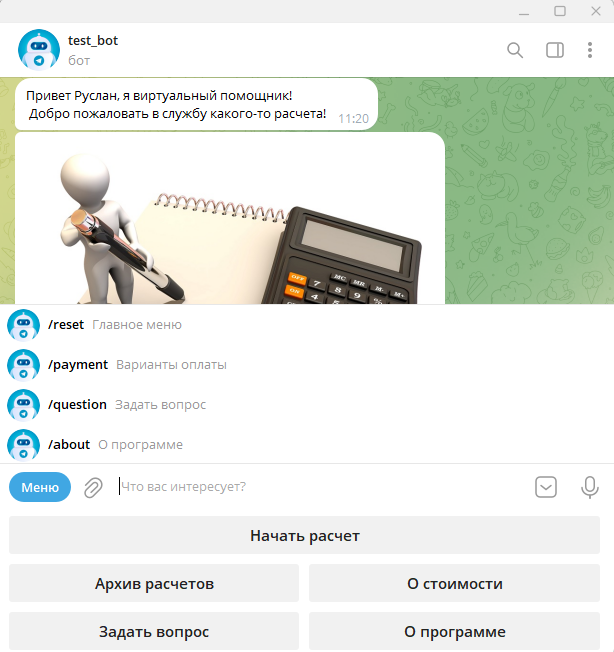
* + «aiogramm 3.9», позволяющая опрашивать бота в асинхронном режиме, что позволяет вести диалог с несколькими пользователями одновременно;
  + «Asyncpg 0.29.0» для работы с базой данных (БД) PostgreSQL. База состоит из 2х таблиц: пользователи и расчеты.
  + «SQLAlchemy 2.0.31» ORM для удобства управлением БД без использования SQL запросов.

Функционал ОРМ и базы данных реализован в файлах папки «database»:

* «models.py» - описание таблиц;
* «engine.py» – взаимодействие БД и ОРМ и создание/удаление таблиц;
* «midlleware.py» - фильтр сессии;
* «orm\_query» - запросы ОРМ к БД.
  + «loguru 0.7.2» основные действия бота и пользователей фиксируются в журнале файла «logger.json». Запись ведется в формате json c сериализацией для дальнейшего анализа данных. Записываемые события:
  + бот запущен;
  + бот остановлен;
  + пользователь зарегистрировался;
  + пользователь создал расчет;
  + пользователь оплатил расчет;
  + пользователь скачал расчет.
  1. Интерфейс пользователя

Весь функционал интерфейса реализован в файле «handlers\user.py»

После регистрации кнопкой «СТАРТ», в БД сохраняются данные пользователя (id\_user/id чата, name) и ему доступны кнопка меню (кнопка реализована программно в файле «commands\bot\_cmd.py») и интерфейс главного меню:



* Начать расчет. Запуск нового расчета с введением необходимых переменных.
* Архив расчетов. Просмотр наличия всех расчетов, в том числе неоплаченных.
* О стоимости (в меню кнопка «Варианты оплаты». Информация об сумме и вариантах оплаты.
* Задать вопрос (в меню кнопка «Задать вопрос». Связь с администратором бота.
* О программе (в меню кнопка «О программе». Информация о программе.
  + 1. Кнопка «Начать расчет»

Для считывания данных из файла «word» с последующим созданием и записью конечного файла применяется библиотека python-docx1.1.2. Работа с файлами «word», а также расчет значений по введенным параметрам осуществляется в файле «temp\_word.py», расположенного в папке «word». Там же находится шаблон в формате «word».

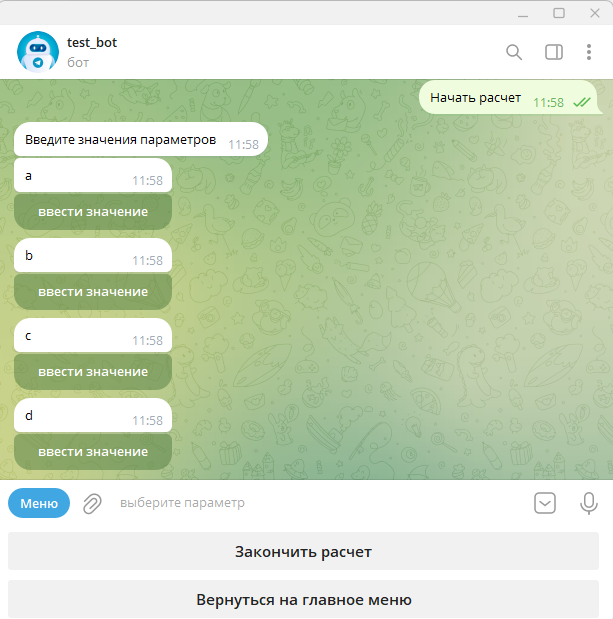
Шаблон файла:

ab = a / b

cd = c \* d

s = ab + cd

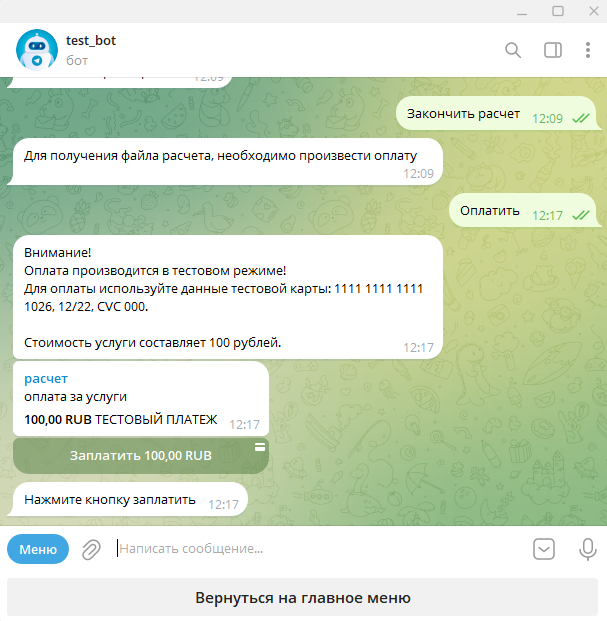
**Примечание:** программа автоматически определяет количество переменных для ввода независимо от количества формул.



На интерфейс выводятся все вводимые параметры для расчета. С помощью inline -кнопок пользователь выбирает необходимый параметр и вводит для него значения

Функции клавиатур описаны в файлах папки «keyboards»

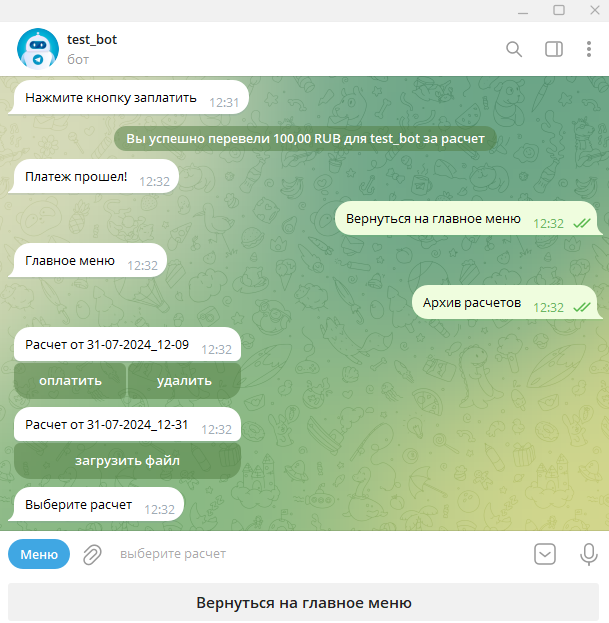
По завершению ввода всех значений (при неполном вводе параметров бот выдаст предупреждение), пользователь нажимает «Закончить расчет» и бот сохраняет произведенный расчет (вернее значение переменных) в БД в таблицу «Calculations». ему предоставляется произвести оплату или вернуться в главное меню.

При оплате предлагается переход на бот оплаты «Telegram». На данный момент подключена тестовая оплата через платежный модуль «Юкасса». Наличие подтверждения платежа проверяется ботом и в БД записываются: дата, данные о сумме и номере платежа.

После оплаты пользователю предлагается скачать файл расчета и бот переходит в главное меню.

Если пользователь перешел в главное меню без оплаты, он может потом оплатить и соответственно скачать файл расчета в интерфейсе «Архив расчетов».

* + 1. Кнопка «Архив расчетов»

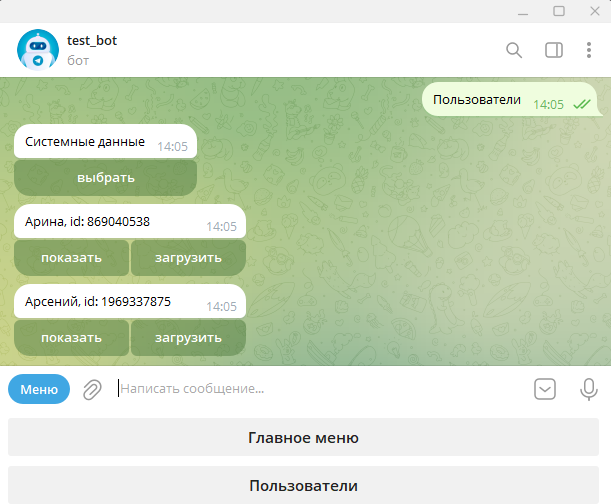
Интерфейс произведенных расчетов, с помощью inline-кнопок предоставляет возможность скачать оплаченный расчет или оплатить для дальнейшего скачивания.

Во избежания конфликта, можно удалять только не оплаченные расчеты.

* 1. Интерфейс администратора.

Весь функционал интерфейса реализован в файле «handlers\admin.py».

* + 1. Чтение журнала логирования



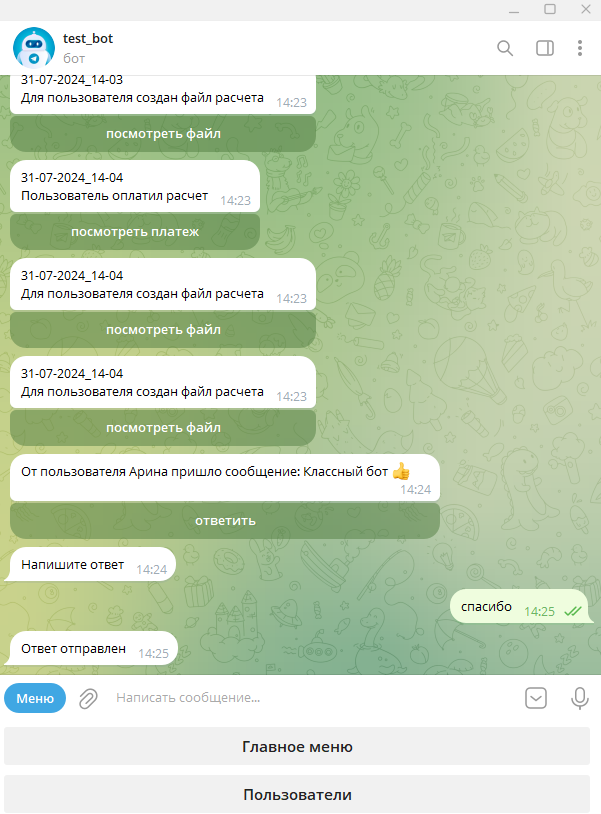
После нажатия кнопки «Пользователи», бот предоставляет список пользователей (в том числе бота – «Системные данные») с inline-кнопками.

Для данных по боту можно только посмотреть события, для пользователей можно выгрузить весь лог его действий в формате «Exel» (для формирования данных используется библиотека «pandas») или посмотреть online.



C помощью inline-кнопок можно посмотреть платеж или посмотреть расчет в файле (файл формируется той же функцией, что и для пользователя).

* + 1. Ответ на вопрос пользователя



При написание пользователем вопроса, к администратору приходит сообщении с inline-кнопкой «ответить». Нажимается кнопка, пишется ответ и отправляется. Бот доставляет ответ пользователю задавшему вопрос.

1. Выводы

Реализованный бот полностью соответствует требования технического задания. Также в боте были реализованы дополнительные возможности, которые позволяют использовать его более функционально.

Приложение 1

Перечень применяемых библиотек.

Общий перечень:

aiofiles==23.2.1  
aiogram==3.9.0  
aiohttp==3.9.5  
aiosignal==1.3.1  
annotated-types==0.7.0  
async-timeout==4.0.3  
asyncpg==0.29.0  
attrs==23.2.0  
certifi==2024.7.4  
charset-normalizer==3.3.2  
click==8.1.7  
colorama==0.4.6  
convtools==1.12.1  
Deprecated==1.2.14  
distro==1.9.0  
et-xmlfile==1.1.0  
frozenlist==1.4.1  
greenlet==3.0.3  
idna==3.7  
jsonschema==4.23.0  
jsonschema-specifications==2023.12.1  
loguru==0.7.2  
lxml==5.2.2  
magic-filter==1.0.12  
multidict==6.0.5  
netaddr==1.3.0  
notifiers==1.3.3  
numpy==2.0.1  
openpyxl==3.1.5  
pandas==2.2.2  
pillow==10.4.0  
pydantic==2.8.2  
pydantic\_core==2.20.1  
python-dateutil==2.9.0.post0  
python-docx==1.1.2  
python-dotenv==1.0.1  
pytz==2024.1  
referencing==0.35.1  
requests==2.32.3  
rpds-py==0.19.1  
six==1.16.0  
SQLAlchemy==2.0.31  
typing\_extensions==4.12.2  
tzdata==2024.1  
urllib3==2.2.2  
win32-setctime==1.1.0  
wrapt==1.16.0  
yarl==1.9.4

Из них были установлены:

aiogram==3.9.0  
 python-docx  
 loguru  
 python-dotenv  
 sqlalchemy  
 asyncpg  
 pandas