# 1. Быстрый старт

Файл config.yaml

bot:

  admin\_id: 10114365

  token: 4666879:Asdfgdfgs6574sdfg6+75sdfg5

database:

  cred\_file: gsheetscred.json

  table\_id: cYbPQrmmVQ9QjhaNUJcALQ

  g\_drive\_media\_folder: c8oIPLZnbj\_0vKX7o

contacts:

admin\_chanel\_id: -10015165464

  new\_user\_chanel\_id: -10015165464

1. Создать бота при помощи **@botfather**
   1. В файле заполнить поле «**token**»
   2. **admin\_id** – телеграм ид пользователя, будет иметь право на команду «**/reload\_users**», которая перечитывает пользователей из таблицы, без перезапуска бота (необходимо если в таблицу пользователей вручную вносились изменения). ИД получить при помощи бота **@username\_to\_id\_bot**
2. Создать сервисный аккаунт Google ([Как создать сервисный аккаунт.mht](file:///C:\Users\29ObrubovVU\AppData\Roaming\Microsoft\Word\Как%20создать%20сервисный%20аккаунт.mht))
   1. Получить файл настроек, записать его в корень проекта, указать его в поле **cred\_file**
3. Создайте таблицу на Google диске
   1. Откройте ее. Из адресной строки взять ее ид и записать в **table\_id**
   2. Нажать кнопку «Настроить доступ» (справа верху) и добавить e-mail сервисного аккаунта (а также e-mail всех пользователей, кто будет с ней работать). Взять из файла, созданного на шаге 2.
   3. Таблица должна содержать листы (названия с маленькой буквы):
      1. «users» с полями «tg\_id, tg\_username, id, fullname, phone, email, stb\_model, mrf, san, mak»
      2. «problems, ideas, questions, change\_user\_data» с полями « mess\_id, user\_id, tg\_username, message, links»
      3. Поля можно назвать по-русски, главное не менять порядок. Таблицу можно форматировать для лучшего восприятия
4. Создайте папку на Google диске, для хранения медиафайлов
   1. Откройте ее. Из адресной строки взять ее ид и записать в **g\_drive\_media\_folder**
   2. Нажать ссылку «Настроить доступ» (справа) и добавить e-mail сервисного аккаунта (а также e-mail всех пользователей, кто будет с ней работать). Взять из файла, созданного на шаге 2.
5. Секция **contacts**
   1. **admin\_chanel\_id** – ИД канала администраторов. Получить при помощи бота **@username\_to\_id\_bot** (сделать бота участником этого канала)
   2. **new\_user\_chanel\_id** – ИД канала для сообщений о новых пользователях (может быть тот же admin\_chanel\_id) . Получить при помощи бота **@username\_to\_id\_bot** (сделать бота участником этого канала)

На машине должен быть установлен Python версии не ниже 3.9. (проверить командой **python3 --version)**

Также установлены библиотеки из файла requirements.txt (установка pip3 install -r requirements.txt)

Запуск бота командой **python3 bot.py**

# 2. Структура проекта

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **app** | **Модули бота** | | | |
|  | * **database** | **Модули работы с Google** | | |
|  |  | \_\_init\_\_.py |  |  |
|  |  | g\_drive\_accessor.py | Работа с Google диском | |
|  |  | g\_spread\_accessor.py | Работа с Google таблицей | |
|  | * **handlers** | **Обработчики сообщений** | | |
|  |  | * **main** | **Диалог Основная работа** | |
|  |  |  | \_\_init\_\_.py |  |
|  |  |  | answers.py | Описание кнопок ответов |
|  |  |  | h\_info.py | Обработчики ветки полезной информации |
|  |  |  | h\_main.py | Обработчики основной ветки диалога |
|  |  |  | messages.py | Сообщения выдаваемые ползователю |
|  |  | * **registration** | **Диалог Регистрация пользователя** | |
|  |  |  | \_\_init\_\_.py |  |
|  |  |  | answers.py | Описание кнопок ответов |
|  |  |  | hendlers.py | Обработчики диалога регистрации |
|  |  |  | messages.py | Сообщения выдаваемые ползователю |
|  |  | \_\_init\_\_.py |  |  |
|  |  | keyboard.py |  | Модуль формирования кнопок ответов |
|  |  | start\_end.py |  | Старт диалога с ботом |
|  | \_\_init\_\_.py |  | | |
|  | create\_bot.py | Процедуры создания объекта бота | | |
|  | globals.py | Глобальные классы и объекты | | |
|  | utils.py | Проверка mac, e-mail | | |
| * **config** | **Модуль чтения файла конфигурации** | | | |
|  | \_\_init\_\_.py |  | | |
|  | config\_reader.py | Модуль чтения файла конфигурации | | |
| * **Docs** | **Документация** | | | |
| .gitignore |  | | | |
| bot.py | Стартовый файл для запуска бота | | | |
| config.yml | Файл конфигурации | | | |
| Dockerfile | Файл для создания Docker контейнера | | | |
| README.md |  | | | |
| requirements.txt | Список необходимых библиотек (установка pip3 install -r requirements.txt) | | | |

# 3. Описание старта работы

## 3.1 **bot.py – стартовый модуль**

При старте модуля выполняется процедура **main**.

Сначала считываются настройки из файла конфигурации заданного в переменной окружения **SETTINGS\_MODULE**, если переменная не задана, то из файла **config.yml**.

Далее создается объект бота и запускается обработка ботом сообщений от пользователей

## 3.2 **app\globals.py** – модуль с глобальными объектами

* **class Bot: -** класс описывающий бота
* **bot** – объект бота
* **class User:** - класс содержащий данные пользователя
  + **\_\_str\_\_(self)** – метод возвращающий полное строковое представление пользователя
  + **main\_info** – возвращает основную информацию по пользователю (отображается в сообщениях в канале админов)
  + **user\_from\_reg\_data** – создает объект пользователя по данным собранным при регистрации нового
  + **user\_from\_g\_table** - создает объект пользователя по данным из Google таблицы (при старте бота)

## 3.3 **app\** **create\_bot.py** – модуль инициализации бота

* **setup\_bot** – инициализация объекта бота
* **setup\_db** – инициализация объектов для работы с хранилищем
* **setup\_aiogram** – инициализация бота aiogram
* **setup\_logging** – инициализация логгирования

## 3.4 **app\** **database\g\_spread\_accessor.py** – модуль работы с Google таблицей

* GoogleDatabase: - класс для работы с Google таблицей
* **\_\_init\_\_** - метод инициализации объекта класса
* **get\_g\_sheets** - получает из таблицы листы по имени
* **load\_users** – загружает данные пользователей из таблицы
* **get\_range\_hyperlink** – получает гиперссылку на диапазон ячеек таблицы
* **get\_row\_from\_range** – получает номер строки из адреса диапазона ячеек
* **write\_user\_to\_db** – записывает в таблицу нового пользователя, возвращает номер строки в таблице с новым пользователем
* **get\_new\_id** – получает новый номер для сообщения записываемого в таблицу
* **write\_user\_request** – записывает в таблицу новое сообщение от пользователя

## 3.5 **app\** **database\g\_drive\_accessor.py** – модуль работы с Google таблицей

* **GoogleDrive:** - класс для работы с Google диском
* **\_\_init\_\_** - метод инициализации объекта класса
* **upload\_files** - скачивает файлы с Телеграма затем вызывает метод их закачки на Гугл диск. Возвращает ид папки на Гугл диске куда сложены файлы
* **upload\_to\_g\_dive** – загружает скачанные файлы на Гугл диск
* **upload\_file\_to\_g\_dive** – выполняет загрузку отдельного файла в отдельном потоке
* **change\_folder\_name** – задает имя папки сообщения по номеру сообщения

# 4. Описание обработки сообщений пользователей

## 4.1 **\app\handlers\start\_end.py** – модуль начала диалога с пользователем

* **cmd\_start** – обработка команды **/start**. Начало диалога с пользователем. Если пользователь зарегистрирован то переходит к диалогу основной работы, иначе к диалогу регистрации
* **cmd\_cancel** – обработчик комманды к началу **/cancel**. Возвращает к началу диалога из любого места.
* **any\_text\_message** – ввод любого сообщения кроме **/start, /cancel, /reload\_users** вызывает стартовый диалог
* **cmd\_reload\_users** – обрабатывает команду **/reload\_users**. Перечитывает список пользователей из таблицы без необходимости перезапуска бота. Необходимо если в список пользователей менялся вручную. Может выполнить только пользователь администратор, указанный в файле **config.yml**
* **register\_handlers\_common** – в этой процедуре определяется, какая команда какой процедурой обрабатывается.

## 4.2 **\app\handlers\registration\hendlers.py**– модуль с обработчиками диалога регистрации

* **registration\_stop** – вызывается при прерывании регистрации
* **registration\_start** – Выдает приветственное сообщение и запрос на согласие на регистрацию
* **registration\_approval** – если согласие на регистрацию получено, запрашивает телефон, иначе прерывает регистрацию
* **registration\_phone** – запоминает телефон, проверяет есть ли у пользователя строковый логин в Телеграм и запрашивает ФИО.
* **registration\_fio** – запоминает ФИО и запрашивает e-mail
* **registration\_email** – проверяет введенный e-mail, если корректый, то запоминает его и запрашивает наличие ТВ-приставки с интерфейсом Wink.
* **registration\_tv** – при положительном ответе запрашивает модель ТВ-приставки. Список моделей приставок задается в константах TV\_BOXES\_1 и TV\_BOXES\_2 в модуле answers.py
* **registration\_model** – запоминает модель выбранной приставки и запрашивает MRF. Список MPF приставок задается в константt MRF в модуле answers.py
* **registration\_mrf** – запоминает MRF и запрашивает SAN.
* **registration\_subscriber** - запоминает SAN и запрашивает MAC.
* **registration\_mak** – проверяет введенный MAC. Записывает пользователя в БД (метод write\_user\_to\_db класса GoogleDatabase: из модуля g\_spread\_accessor.py). Отправляет в чат новых пользователей сообщение о новом пользователе. Выдает сообщение об окончании регистрации. И возвращается к начальному диалогу.

## 4.3 **\app\handlers\main\h\_main.py**– модуль с обработчиками основного диалога

* **main\_start** – старт основного диалога, ожидание ввода основных команд.
* **handle\_main\_command** – обработка основных комманд.
  + **"Сообщить о проблеме", "Предложить идею", "Задать вопрос"** – запрашивает ввод информации для передачи команде Wink
  + **"Чек-лист"** – выводит текстовое сообщение
  + **"Полезная информация"** - переход к меню полезной информации (\app\handlers\main\h\_info.py info\_start)
* **handle\_input\_info –** обработка введенной для отправки информации. При вводе текстовой или медиа информации данные запоминаются и ожидается дальнейший ввод информации (если медиа файл превышает 20 Мб, выдается предупреждение). При нажатии кнопки «Отправить…» сохраненные данные записываются в таблицу БД, медиафайлы сохраняются на Гугл диск и сообщение перенаправляется в чат админов.
* **register\_handlers\_main** – в этой процедуре определяется, какая команда какой процедурой обрабатывается.

## 4.3 **\app\handlers\main\h\_** **info.py**– модуль с обработчиками диалога полезной информации

* **info\_start** – старт диалога, ожидание ввода команд.
* **handle\_** **info\_command** – обработка команд меню информации.
  + **"Мои данные"**– выводит данные пользователя и предлагает ввести запрос на изменение данных.
  + **"Как пользоваться ботом", "Вопросы и ответы", "Дайджесты"**– выводит текстовое сообщение
  + **"Главное меню"**- возврат в главное меню
* **handle\_user\_info** – если выбран запрос на изменение данных, ожидает ввода запроса, иначе возврат к меню информации.
* **handle\_change\_user\_info** – обрабатывает запрос на изменение информации. Записывает запрос в БД. Пересылает сообщение в чат администраторов. И Возвращается к меню информации
* **register\_handlers\_info** – в этой процедуре определяется, какая команда какой процедурой обрабатывается.