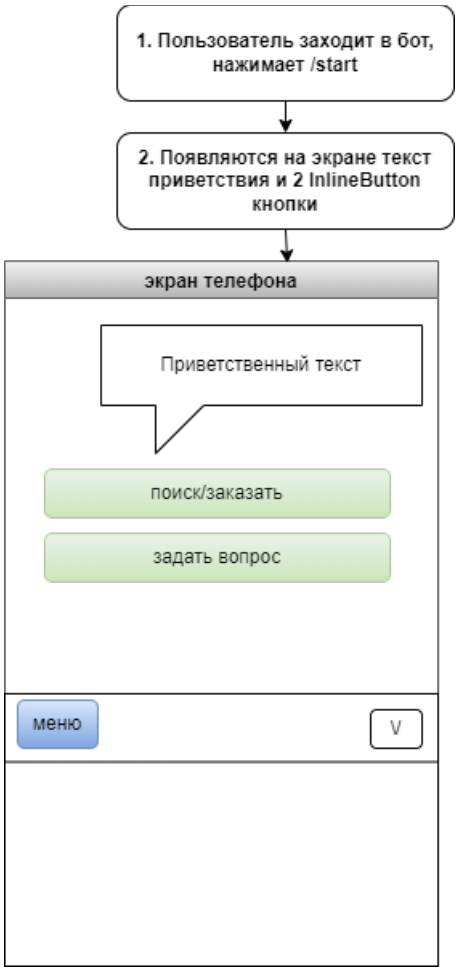
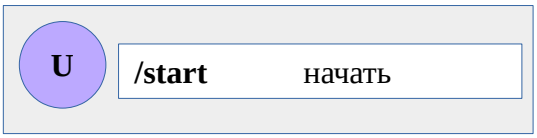
Тз телеграм бот.

Надо:

язык разработки: Python >=3.9;
база данных: MySql;
работа с бд в коде: SQL запросы или SQLAlchemy — на усмотрение;
фреймворк: Aiogram 3, либо что-то на свое усмотрение;
хранение секретных данных необходимых приложению: файл *.ENV* либо *config.py;*
requirements.txt: да;
github + .gitignore: да;
документирование кода на русском;
приложение должно быть пригодным для расширения;



Кнопка: меню



По нажатии /start появляются кнопки и создается таблица в базе данных если ее не существует:

```
# @params { tg_usid: str } - id пользователя телеграм
# @return { sql: str } - строка запроса

@classmethod
def get_sql_new_basket(cls, tg_usid: str) -> str:
       sql: str = f"CREATE TABLE IF NOT EXISTS `e110kw29_kitopt`.`{'User_'+tg_usid}`
       ( `id` INT NOT NULL AUTO_INCREMENT , `product_id` TEXT CHARACTER SET utf8 COLLATE
       utf8_general_ci NOT NULL, `photo` TEXT CHARACTER SET utf8 COLLATE utf8_general_ci
       NOT NULL, `user_id` TEXT CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL ,
       `user_name` TEXT CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL ,
       `uniq_token` TEXT CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL , `name`
       TEXT CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL , `datetime` TEXT
       CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL , `category` TEXT CHARACTER
       SET utf8 COLLATE utf8_general_ci NOT NULL , `data_type` TEXT CHARACTER SET utf8
       COLLATE utf8_general_ci NOT NULL , `brand` TEXT CHARACTER SET utf8 COLLATE
       utf8_general_ci NOT NULL , `name_good` TEXT CHARACTER SET utf8 COLLATE
       utf8_general_ci NOT NULL , `characteristic` TEXT CHARACTER SET utf8 COLLATE
       utf8_general_ci NOT NULL , `count_on_stock` INT NOT NULL , `count_on_order` INT
       NOT NULL , `price_from_1to2` DOUBLE NOT NULL , `price_from_3to4` DOUBLE NOT NULL ,
       `price_from_5to9` DOUBLE NOT NULL , `price_from_10to29` DOUBLE NOT NULL ,
       `price_from_30to69` DOUBLE NOT NULL , `price_from_70to149` DOUBLE NOT NULL ,
       `price_from_150` DOUBLE NOT NULL , `sum_position` DOUBLE NOT NULL ,
       `delivery_method` TEXT CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL ,
       `price_delivery` DOUBLE NOT NULL , `address_delivery` TEXT CHARACTER SET utf8
       COLLATE utf8_general_ci NOT NULL , `time_delivery` TEXT CHARACTER SET utf8 COLLATE
       utf8_general_ci NOT NULL , `pay_method` TEXT CHARACTER SET utf8 COLLATE
       utf8_general_ci NOT NULL , `pay_status` TEXT CHARACTER SET utf8 COLLATE
       utf8_general_ci NOT NULL , `order_status` TEXT CHARACTER SET utf8 COLLATE
       utf8_general_ci NOT NULL , PRIMARY KEY (`id`)) ENGINE = InnoDB;"
     return sql
```

Эта таблица — **корзина пользователя**. Имя таблицы: User_{id пользователя телеграм} (например: User_123456789)

Далее запускается асинхронный таймер на определенное время (24 часа). Таймер нужен чтобы удалять таблицы корзин пользователей после окончания работы с ботом и не засирать базу данных. На одного пользователя создается одна таблица корзины и запускается один таймер. При многократном перезапуске бота путем нажатий **/start**, новых таймеров создаваться не должно для этого пользователя, также как не должно создаваться новой таблицы (`CREATE TABLE IF NOT EXISTS …`) .

Пример таймера:

```
from aio_timers import Timer  # pip install aio-timers
from typing import Callable

async def run_sesion_timer(self, params: dict = None):
    """
    params = {
      id: id_tg:uniq_token
      time_session: int,
      callback: Callable,
      args: *args,

    }
    """

    try:
      time_session = 5000  # секунды
      timer = None
      if params:
        id = params.get('id')
        time_sesion = int(params.get('time_session'))
        callback = params.get('callback')
        args = params.get('args', tuple())
        if id and (isinstance(callback, Callable)) and len(args) > 0:
          timer = Timer(time_sesion, callback, callback_args=args, callback_async=True)
          if not self.curr_timers.get(id):
            self.curr_timers[id] = timer
        else:
          raise Exception("\nmethod: run_session_timer =>\n---error arguments\n")
    except Exception as e:
      print(e)
    else:
      if timer:
        await timer.wait()
```

После того как таймер отработал, нужно удалить таблицу (корзину пользователя) полностью из базы данных и **отправить сообщение** что корзина очищена, даже ели она не пуста.

```
@classmethod
  def get_sql_delete_table(cls, id_user: str) -> str:
    sql: str = f"DROP TABLE {'User_'+id_user};"
    return sql
```

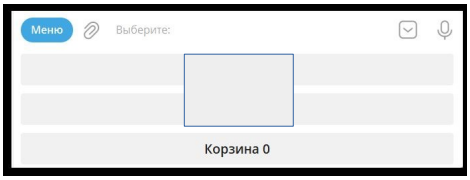
В примере выше для этого в конструктор Timer

```
    timer = Timer(time_sesion, callback, callback_args=args, callback_async=True)
```

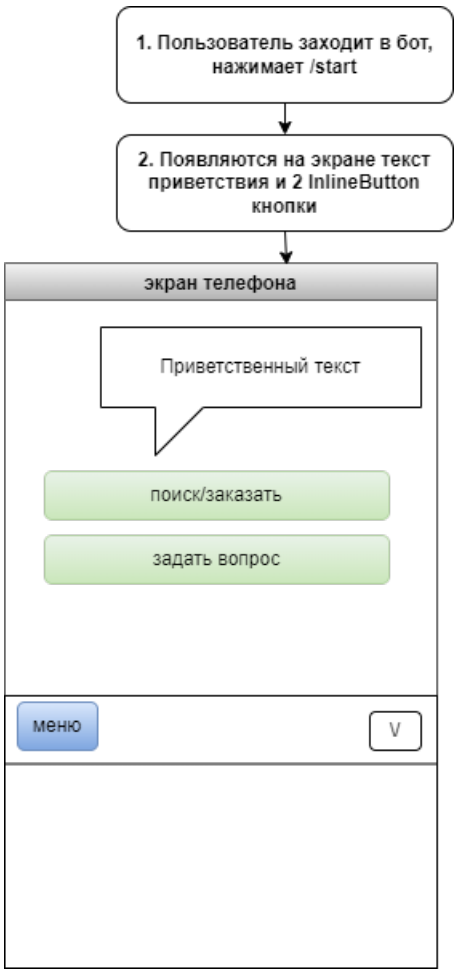
передается колбек-метод реализующий функционал удаления таблицы (корзины пользователя)

После создания корзины, вместо стандартной клавиатуры должно появится кнопка

```
    types.KeyboardButton(text=f"Корзина {str(count or 0)}")
    # count - кол-во заказов — корзине на текущий момент
```



Теперь:



По нажатии на кнопке **поиск/заказать** открывается веб-приложение:

```
[types.InlineKeyboardButton(text="поиск/заказать", web_app=WebAppInfo(url=f"https://web-app-bot-b.vercel.app/?usid=User_{usid}"))]
```

в url передается query параметр: usid=User_{id пользователя телеграм} (например: `usid=User_123456789`) Веб-приложение получает get запрос с параметром названия таблицы корзины пользователя в бд, отдает html страницу, пользователь выбирает товар, добавляет/удаляет в/из корзины.

Бот и веб-приложение имеют доступ к одной и той же базе данных

Когда пользователь определился с выбором, он нажимает в веб-приложении кнопку [сделать заказ], веб-приложение закрывается.

Веб-приложение закрывается:

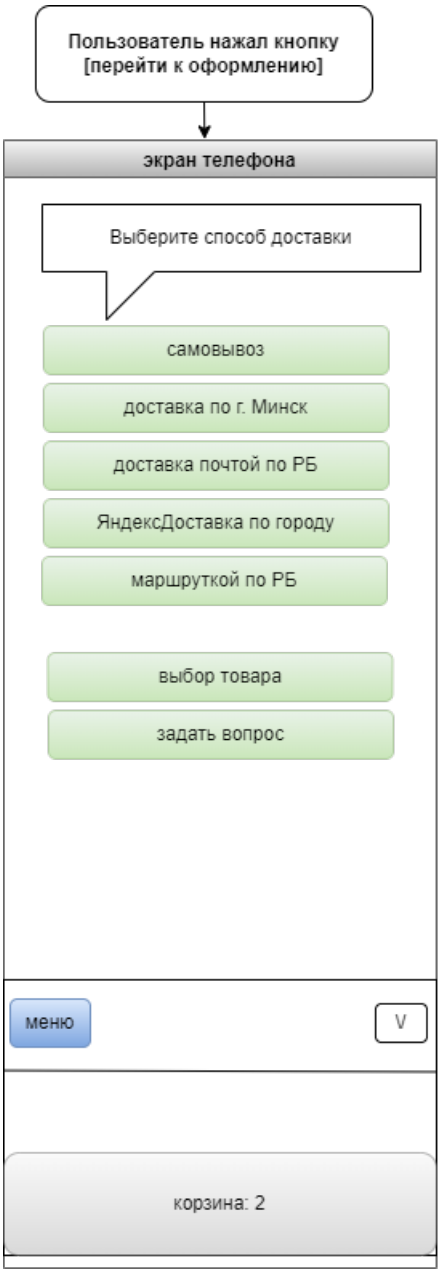
После нажатия кнопки [сделать заказ] в веб-приложении, происходит запрос боту из веб-приложения на скрытие встроенного браузера в телеграм, и вытягивании таблицы корзины юзера из бд. Обновляется кол-во заказов на кнопке [корзина]. По нажатии на кнопке [корзина] появляется сообщение :

- товар А 1 шт 12,5 руб.
- товарБ 15 шт 4 руб.

итого: 16,5 руб.

Это сообщение с кнопками появляется сразу после выхода из веб-приложения, а также по нажатии на кнопку [корзина]

кнопка [перейти к оформлению]


Пользователь нажал кнопку
[перейти к оформлению]
экран телефона
Выберите способ доставки
самовывоз
доставка по г. Минск
доставка почтой по РБ
ЯндексДоставка по городу
маршруткой по РБ
выбор товара
задать вопрос
меню
V
корзина: 2

Нажал кнопку Самовывоз:

Сообщение:

- прикрепить адрес, фото, схему
- стоимость доставки
- сроки отгрузки: в день заказа

нажал кнопку Доставка по г. Минск:

сообщение:

- стоимость доставки: *выгрузить из бд стоимость доставки*
- сроки отгрузки: при оформлении до 16:30 доставка в день оформления

нажал кнопку Доставка почтой по РБ:

сообщение:

кнопка Европочта:

сообщение:

- Европочта
- сроки отгрузки: при оформлении до 16:30 отправка в день оформления
- 100 % предоплата

кнопка Автолайт (от 100 единиц):

сообщение:

- Автолайт (от 100 единиц)
- стоимость доставки:
- сроки отгрузки: на следующий день
- 100% предоплата

нажал кнопку Маршруткой по РБ:

сообщение:

- стоимость доставки: 5-10 руб
- сроки отгрузки: доставка на завтра
- 100% предоплата

нажал кнопку ЯндексДоставка по городу:

сообщение:

- стоимость доставки: по тарифу Яндекс
- сроки отгрузки: через 10 мин.
- 100% предоплата

После выбора доставки помимо описания есть 2 кнопки: 'ок, продолжить' и 'изменить способ доставки'(возврат к способам доставки и последующея перезапись в таблице заказа User_tgID)

кнопка [ok, продолжить]

- Ввод пользователем адреса доставки и запись в бд…
  - **адрес полный**: *если доставка по минску или РБ любым из способов, кроме самовывоза плюс возможность ввести комментарий*
  - **дата доставки**: *если время до 15:30 по мск - есть кнопка сегодня. в противном случае есть кнопки 'завтра' и далее по датам* . При самовывозе Дата и Время когда будет самовывоз
  - **Время доставки:** *выбрать кнопками из диапозонов (с 16:00 до 18:00, с 18:00 до 20:00, с 20:00 до 22:00 и прочее)*

Вывести сообщением 'Проверьте всё ли верно: Вы заказали.. 'Корзина', Способ доставки, адрес, дата, время// кнопки 'Верно, к выбору способа оплаты' 'Изменить заказ' 'Изменить адрес' 'Изменить дату и время доставки'

кнопка [Верно, к выбору способа оплаты]

кнопки

Способы оплаты

Самовывоз

Курьеру при получении

Безналичный расчет БНБ банк/реквизиты карты/сбросить платёжку боту

yoomoney зарегить/после оплаты сообщение USERу о подтверждении оплаты, оформление

'Отправить на комплектовку на склад'

Благодарю за заказ! Передамим как договаривались :)

Подтверждённый заказ улетает менеджеру из списка

Все вводимые пользователем данные записываются (sql: UPDATE) в таблицу корзины. После завершения процедуры оформления все данные из корзины записываются в таблицу заказов, далее уничтожается таблица корзины пользователя, уничтожается таймер.