Додаток

Main.py

```
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
import asyncio
import logging
from Module.Bot import settings
from aiogram import Bot, Dispatcher
from aiogram.enums.parse_mode import ParseMode
from aiogram.fsm.storage.memory import MemoryStorage
from Module.Bot.handlers import router


async def main():
    bot = Bot(token=settings.API_TOKEN, parse_mode=ParseMode.HTML)
    dp = Dispatcher(storage=MemoryStorage())
    dp.include_router(router)
    await bot.delete_webhook(drop_pending_updates=True)
    await dp.start_polling(bot, allowed_updates=dp.resolve_used_update_types())


if __name__ == "__main__":
    logging.basicConfig(level=logging.INFO)
    asyncio.run(main())
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

handlers.py

```
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
from aiogram import F, Router, types
from aiogram.filters import Command, Text
from aiogram.types import Message
from aiogram import flags
from aiogram.fsm.context import FSMContext
from aiogram.fsm.state import State
from aiogram.methods import SendMessage, GetFile
from aiogram.utils.keyboard import ReplyKeyboardBuilder
from Module.Tools.states import *
import Module.Bot.kb as kb
from Module.Tools.filters import *
from aiogram.types.callback_query import CallbackQuery
from Module.Tools import SQLite
import Module.Tools.FilePath as FilePath
from Module.Tools.text import *
from aiogram.enums import ChatType

router = Router()
list_id = []
media = []


@router.message(Command("start"), F.chat.type == ChatType.PRIVATE)
async def start(msg: Message):
    await msg.answer(welcome_message, reply_markup=kb.start_panel)
    try:
        SQLite.writeInfo(FilePath.people,
```

```
                """INSERT INTO users (user_id,name) VALUES (?,?)""",
                (msg.from_user.id, msg.from_user.first_name))
    except:
        pass


@router.message(Command("admin"), IsAdmin())
async def adminm(msg: Message):
    await msg.answer(text="1", reply_markup=kb.admin_panel)


@router.message(F.text == "Відправити повідомлення", IsAdmin())
async def AdmNotification(msg: Message, state: FSMContext):
    await state.set_state(Notification.nf)
    await msg.answer("Напишіть повідомлення")


@router.message(F.text == "Відміна", IsAdmin())
async def AdminCancel(msg: Message, state: FSMContext):
    await msg.answer(text="Повернення до головного меню",
                     reply_markup=kb.admin_panel)
    await state.clear()


@router.message(Notification.nf)
async def processNotifiacationAll(msg: Message, state: FSMContext):
    await state.update_data(nf=msg)
    temp = SQLite.readInfo(FilePath.people, """SELECT user_id from users""")
    print(temp)
    for i in temp:
        try:
            await SendMessage(chat_id=i[0], text=msg.text)
        except:
            pass
    await state.clear()


@router.message(F.text == "Розклад")
async def get_schedule(msg: Message, state: FSMContext):
    await state.set_state(grp.choicegrp)
    await msg.answer(text='Оберіть спеціальність', reply_markup=kb.choice_prof)


@router.callback_query(Text("monday"))
async def monday(callback: CallbackQuery):
    usrid = callback.from_user.id
    grp = SQLite.readInfo(
        FilePath.people,
        f"SELECT own_group FROM users WHERE user_id = '{usrid}'")[0][0]
    sc = SQLite.readInfo(FilePath.schedule,
                         f"SELECT Понеділок FROM {grp}")[0][0]
    await callback.answer(text=str(sc), show_alert=True)
```

```
@router.callback_query(Text("tuesday"))
async def tuesday(callback: CallbackQuery):
    usrid = callback.from_user.id
    grp = SQLite.readInfo(
        FilePath.people,
        f"SELECT own_group FROM users WHERE user_id = '{usrid}'")[0][0]
    sc = SQLite.readInfo(FilePath.schedule,
                         f"SELECT Вівторок FROM {grp}")[0][0]
    await callback.answer(text=str(sc), show_alert=True)


@router.callback_query(Text("wednesday"))
async def wednesday(callback: CallbackQuery):
    usrid = callback.from_user.id
    grp = SQLite.readInfo(
        FilePath.people,
        f"SELECT own_group FROM users WHERE user_id = '{usrid}'")[0][0]
    sc = SQLite.readInfo(FilePath.schedule, f"SELECT Середа FROM {grp}")[0][0]
    await callback.answer(text=str(sc), show_alert=True)


@router.callback_query(Text("thursday"))
async def thursday(callback: CallbackQuery):
    usrid = callback.from_user.id
    grp = SQLite.readInfo(
        FilePath.people,
        f"SELECT own_group FROM users WHERE user_id = '{usrid}'")[0][0]
    sc = SQLite.readInfo(FilePath.schedule, f"SELECT Четвер FROM {grp}")[0][0]
    await callback.answer(text=str(sc), show_alert=True)


@router.callback_query(Text("friday"))
async def friday(callback: CallbackQuery):
    usrid = callback.from_user.id
    grp = SQLite.readInfo(
        FilePath.people,
        f"SELECT own_group FROM users WHERE user_id = '{usrid}'")[0][0]
    sc = SQLite.readInfo(FilePath.schedule, f"SELECT Пятниця FROM {grp}")[0][0]
    await callback.answer(text=str(sc), show_alert=True)


@router.message(F.text == 'Вступникам')
async def matriculant(msg: Message):
    await msg.answer("Оберіть пункт меню", reply_markup=kb.matriculant_panel)


@router.message(F.text == 'Спеціальності')
async def profession(msg: Message):
    await msg.answer("Оберіть спеціальність", reply_markup=kb.profession_panel)
```

```
@router.message(
    F.text.in_({
        'АТ - 274', 'ЕТ - 141', 'ЕП - 051', 'КІ - 123', 'КН - 122', 'ТТ - 275',
        'ГМ - 133'
    }))
async def choiceProf(msg: Message):
    match msg.text:
        case 'АТ - 274':
            await msg.answer(prof_at)
            await msg.answer(info_at)
        case 'ЕТ - 141':
            await msg.answer(prof_et)
            await msg.answer(info_et)
        case 'ЕП - 051':
            await msg.answer(prof_ep)
            await msg.answer(info_ep)
        case 'КІ - 123':
            await msg.answer(prof_ki)
            await msg.answer(info_ki)
        case 'КН - 122':
            await msg.answer(prof_kn)
            await msg.answer(info_kn)
        case 'ТТ - 275':
            await msg.answer(prof_tt)
            await msg.answer(info_tt)
        case 'ГМ - 133':
            await msg.answer(prof_gm)
            await msg.answer(info_gm)


@router.message(F.text == "Перелік документів")
async def doclist(msg: Message):
    await msg.answer(documents_list)
    await msg.answer(warn, parse_mode="MarkdownV2")


@router.message(F.text == "Прийом документів")
async def getdocs(msg: Message):
    await msg.answer(docs9, parse_mode="MarkdownV2")


@router.message(F.text == "Результати прийому")
async def send_result(msg: Message):
    await msg.answer(result1, parse_mode="MarkdownV2")
    await msg.answer(result2, parse_mode="MarkdownV2")
    await msg.answer(result3, parse_mode="MarkdownV2")


@router.message(F.text == "< Назад")
async def nz(msg: Message):
    await msg.answer("Повернення", reply_markup=kb.matriculant_panel)
```

```
@router.message(F.text == "<< Назад")
async def nz1(msg: Message):
    await msg.answer("Повернення", reply_markup=kb.start_panel)


@router.message(F.text == "Абітурієнтам")
async def abi(msg: Message):
    await msg.answer(text="Оберіть спеціальність", reply_markup=kb.choice_prof)


@router.message(F.text.in_({'АТ', 'ЕТ', 'ЕП', 'КІ', 'КН', 'ТТ', 'ТМ'}))
async def getgr(msg: Message, state: FSMContext):
    await state.set_state(grp.choicegrp)
    builder = ReplyKeyboardBuilder()
    temp = twentyfive(msg.text)
    for i in temp:
        builder.add(types.KeyboardButton(text=str(i)))
    builder.adjust(4)
    builder.row(types.KeyboardButton(text="Головне меню"))
    await msg.answer("Оберіть групу",
                     reply_markup=builder.as_markup(resize_keyboard=True))


@router.message(grp.choicegrp)
async def chgrp(msg: Message, state: FSMContext):
    await state.update_data(choicegrp=msg.text)
    SQLite.writeInfo(FilePath.people,
                     f"""UPDATE users SET own_group = ? WHERE user_id = ?""",
                     (msg.text, msg.from_user.id))
    await msg.answer("Оберіть день тижня", reply_markup=kb.schedule_panel)
    await msg.answer(
        "Повернення до головного меню\n(Для того щоб знов подивитися розклад достатньо
відправити /schedule)",
        reply_markup=kb.start_panel)
    await state.clear()


def twentyfive(val):
    res = SQLite.readInfo(
        FilePath.schedule, """select * from sqlite_master
            where type = 'table'""")
    temp = list()
    for i in res:
        if val in i[1]:
            temp.append(i[1])
    return temp


@router.message(F.text == "Головне меню")
async def rettostrt(msg: Message):
```

```
    await msg.answer("Повернення до головного меню",
                     reply_markup=kb.start_panel)


@router.message(F.text == "Приклад документів")
async def sendExampledoc(msg: Message):
    await msg.answer_photo(
        photo=types.FSInputFile("photos/svidotstvo.jpg"),
        caption="Приклад документу про здобутий освітній рівень")
    await msg.answer_photo(photo=types.FSInputFile("photos/passport.jpg"),
                           caption="Приклад пасспорту")
    await msg.answer_photo(photo=types.FSInputFile("photos/inn.png"),
                           caption="Приклад ідентифікаційного коду")
    await msg.answer_photo(photo=types.FSInputFile("photos/viys.png"),
                           caption="Приклад військово облікових документів")
    await msg.answer_photo(photo=types.FSInputFile("photos/zno.jpg"),
                           caption="Приклад сертифікату зно")
    await msg.answer_photo(photo=types.FSInputFile("photos/forma_086-1.jpg"),
                           caption="Приклад форми 086-1/о ")
    await msg.answer_photo(photo=types.FSInputFile("photos/forma-063-o.jpg"),
                           caption="Приклад форми 063/о")


@router.message(Command("schedule"))
async def getsch(msg: Message):
    val = SQLite.readInfo(
        FilePath.people,
        f"SELECT own_group from users WHERE user_id == {msg.from_user.id}")
    twentyfive(val[0][0])
    await msg.answer("Оберіть день тижня", reply_markup=kb.schedule_panel)
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
kb.py
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
from aiogram.types import InlineKeyboardButton, InlineKeyboardMarkup, KeyboardButton,
ReplyKeyboardMarkup, ReplyKeyboardRemove

admin_panel = [
    [
        KeyboardButton(text="Відправити повідомлення"),
    ],
]
admin_panel = ReplyKeyboardMarkup(keyboard=admin_panel, resize_keyboard=True)

admin_send_notification = [[
    KeyboardButton(text="Відміна"),
]]

admin_send_notification = ReplyKeyboardMarkup(keyboard=admin_send_notification,
                                              resize_keyboard=True)

student_panel = [[
    KeyboardButton(text="Розклад"),
```

```
]]
student_panel = ReplyKeyboardMarkup(keyboard=student_panel,
                                    resize_keyboard=True)

cancel = [[KeyboardButton(text="Відміна")]]
cancel = ReplyKeyboardMarkup(keyboard=cancel, resize_keyboard=True)

schedule_panel = [[
   InlineKeyboardButton(text="Понеділок", callback_data="monday"),
   InlineKeyboardButton(text="Вівторок", callback_data="tuesday"),
   InlineKeyboardButton(text="Середа", callback_data="wednesday"),
   InlineKeyboardButton(text="Четвер", callback_data="thursday"),
   InlineKeyboardButton(text="П'ятниця", callback_data="friday"),
]]
schedule_panel = InlineKeyboardMarkup(inline_keyboard=schedule_panel)

start_panel = [[
   KeyboardButton(text='Вступникам'),
   KeyboardButton(text='Абітурієнтам'),
]]
start_panel = ReplyKeyboardMarkup(keyboard=start_panel, resize_keyboard=True)

matriculant_panel = [[KeyboardButton(text='Спеціальності')],
                     [
                        KeyboardButton(text='Перелік документів'),
                        KeyboardButton(text='Приклад документів')
                     ], [KeyboardButton(text='Прийом документів')],
                     [KeyboardButton(text='Результати прийому')],
                     [KeyboardButton(text="<< Назад")]]

matriculant_panel = ReplyKeyboardMarkup(keyboard=matriculant_panel,
                                        resize_keyboard=True)

profession_panel = [[
   KeyboardButton(text='АТ - 274'),
   KeyboardButton(text='ЕТ - 141'),
   KeyboardButton(text='ЕП - 051'),
   KeyboardButton(text='КІ - 123'),
   KeyboardButton(text='КН - 122'),
   KeyboardButton(text='ТТ - 275'),
   KeyboardButton(text='ГМ - 133'),
], [KeyboardButton(text="< Назад")]]

profession_panel = ReplyKeyboardMarkup(keyboard=profession_panel,
                                       resize_keyboard=True)

choice_prof = [[
   KeyboardButton(text='АТ'),
   KeyboardButton(text='ЕТ'),
   KeyboardButton(text='ЕП'),
   KeyboardButton(text='КІ'),
   KeyboardButton(text='КН'),
```

```
    KeyboardButton(text='ТТ'),
    KeyboardButton(text='ГМ'),
], [KeyboardButton(text="Головне меню")]]

choice_prof = ReplyKeyboardMarkup(keyboard=choice_prof, resize_keyboard=True)
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
settings.py
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
API_TOKEN = '5814746272:AAHlPJ79vgXvu1pvjU5pmZK4ER4AIJ8dNmo'
AdminList = ['607445709']
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
FilePath.py
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
people = "DataBase/people.db"
schedule = "DataBase/schedule.db"
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
Filters.py
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
from aiogram import types
from aiogram.filters import Filter
from Module.Bot.settings import AdminList
from Module.Tools import SQLite
from Module.Tools import FilePath


class IsAdmin(Filter):

    async def __call__(self, message: types.Message):
        return str(message.from_user.id) in AdminList
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
SQLite.py
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
import sqlite3


def readInfo(Path: str, Request: str):
    with sqlite3.connect(Path) as Base:
        Method = Base.cursor()

        try:
            Method.execute(Request)
        except Exception as Except:
            print(f"[SQL]> {Except}")

        return Method.fetchall()


def writeInfo(Path: str, Request: str, Val: set = None):
    with sqlite3.connect(Path) as Base:
        Method = Base.cursor()
        Method.execute(Request, Val)
\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
```

States.py
\\

```
from aiogram.fsm.state import StatesGroup, State


class grp(StatesGroup):
    choicegrp = State()


class Notification(StatesGroup):
    nf = State()
```

\\\

Text.py
\\

```
welcome_message = "Вітаю я бот Фахового коледжу транспорту на комп'ютерних технологій"

prof_at = "АТ – 274 “Автомобільний транспорт” (освітня програма “Обслуговування та ремонт автомобілів і двигунів”)"
prof_et = "ЕТ – 141 “Електроенергетика, електротехніка та електромеханіка” (освітня програма “Обслуговування і ремонт електроустаткування автомобілів і тракторів”)"
prof_ep = "ЕП – 051 “Економіка” (освітня програма “Економіка підприємства”)"
prof_ki = "КІ – 123 “Комп’ютерна інженерія” (освітня програма “Обслуговування комп’ютерних систем і мереж”)"
prof_kn = "КН – 122 “Комп’ютерні науки” (освітня програма “Обслуговування програмних систем і комплексів”)"
prof_tt = "ТТ – 275 “Транспортні технології (на автомобільному транспорті)” (освітня програма “Організація перевезень і управління на автотранспорті”)"
prof_gm = "ГМ – 133 “Галузеве машинобудування” (освітня програма “Комп’ютерний інжиніринг в машинобудуванні”)"

info_at = "На денному відділенні \n• На базі 9 класів: \n3 роки 10 місяців \n• На базі 11 класів: \n2 роки 10 місяців \nНа заочному відділенні \n• На базі 11 класів: \n3 роки 3 місяці"
info_et = "На денному відділенні \n• На базі 9 класів: \n3 роки 10 місяців \n• На базі 11 класів: \n2 роки 10 місяців"
info_ep = "На денному відділенні \n• На базі 9 класів: \n2 роки 10 місяців \n• На базі 11 класів: \n1 рік 10 місяців"
info_ki = "На денному відділенні \n• На базі 9 класів: \n3 роки 10 місяців \n• На базі 11 класів: \n2 роки 10 місяців \nНа заочному відділенні \n• На базі 11 класів: \n3 роки 3 місяці"
info_kn = "На денному відділенні \n• На базі 9 класів: \n3 роки 10 місяців"
info_tt = "На денному відділенні \n• На базі 9 класів: \n3 роки 10 місяців \n• На базі 11 класів: \n2 роки 10 місяців"
info_gm = "На денному відділенні \n• На базі 9 класів: \n3 роки 10 місяців"

documents_list = "• КОПІЯ документу про здобутий освітній (освітньо-кваліфікаційний) рівень\n• КОПІЯ документу, що посвідчує особу (копія ID-паспорта)\n• КОПІЯ ідентифікаційного коду\n• КОПІЇ військово-облікових документів (при наявності)\n• СЕРТИФІКАТ ЗНО, НМТ (для вступників на основі 11 класів)\n• кольорову фотокартку розміром 3х4 см в ЕЛЕКТРОННІЙ формі (у вигляді ФАЙЛУ розміром до 1 Мб)\n• МЕДИЧНІ ДОВІДКИ за формою 086-1/о (для фізвиховання після зарахування) та 063-о (про щеплення), флюорографія\n• інші КОПІЇ документів подаються вступником, якщо це
```

викликано особливими умовами вступу на відповідні конкурсні пропозиції, установлені законодавством, у строки, визначені для прийому документів"
warn = "*Увага\! Копії документів без пред’явлення оригіналів не приймаються*"
docs9 = "*• Після 9 класів на денну форму навчання*\n з 30 червня по 13 липня \n\n*• Після 11 класів на денну форму навчання *\nз 10 липня по 31 липня для осіб, що вступають на основі індивідуальної усної співбесіди \nз 10 липня по 9 серпня для осіб, що подають сертифікати ЗНО, НМТ \n\n*• На заочну форму навчання* \n *з 9 жовтня по 20 жовтня*"

result1 = "Результати індивідуальної усної співбесіди з української мови та математики:
[ТУТ](https://drive.google.com/drive/folders/1Vv0zKlT85l8730Tk6cR9DbsUvgXZmVwb)"
result2 = "Списки рекомендованих на бюджет \(після 9 класів\):
[ТУТ](https://drive.google.com/drive/folders/172WmsSdwvJCFFg-YN0CkfJ4AqEjzBbGx)"
result3 = "Накази на зарахування:
[ТУТ](https://drive.google.com/drive/folders/1vqefyJox5mnNQa1rrVWS6Vn95KVFsfPZ)"
\\\