DnevnikBot

Telegram API

Мирзоев Магомед

ВВЕДЕНИЕ

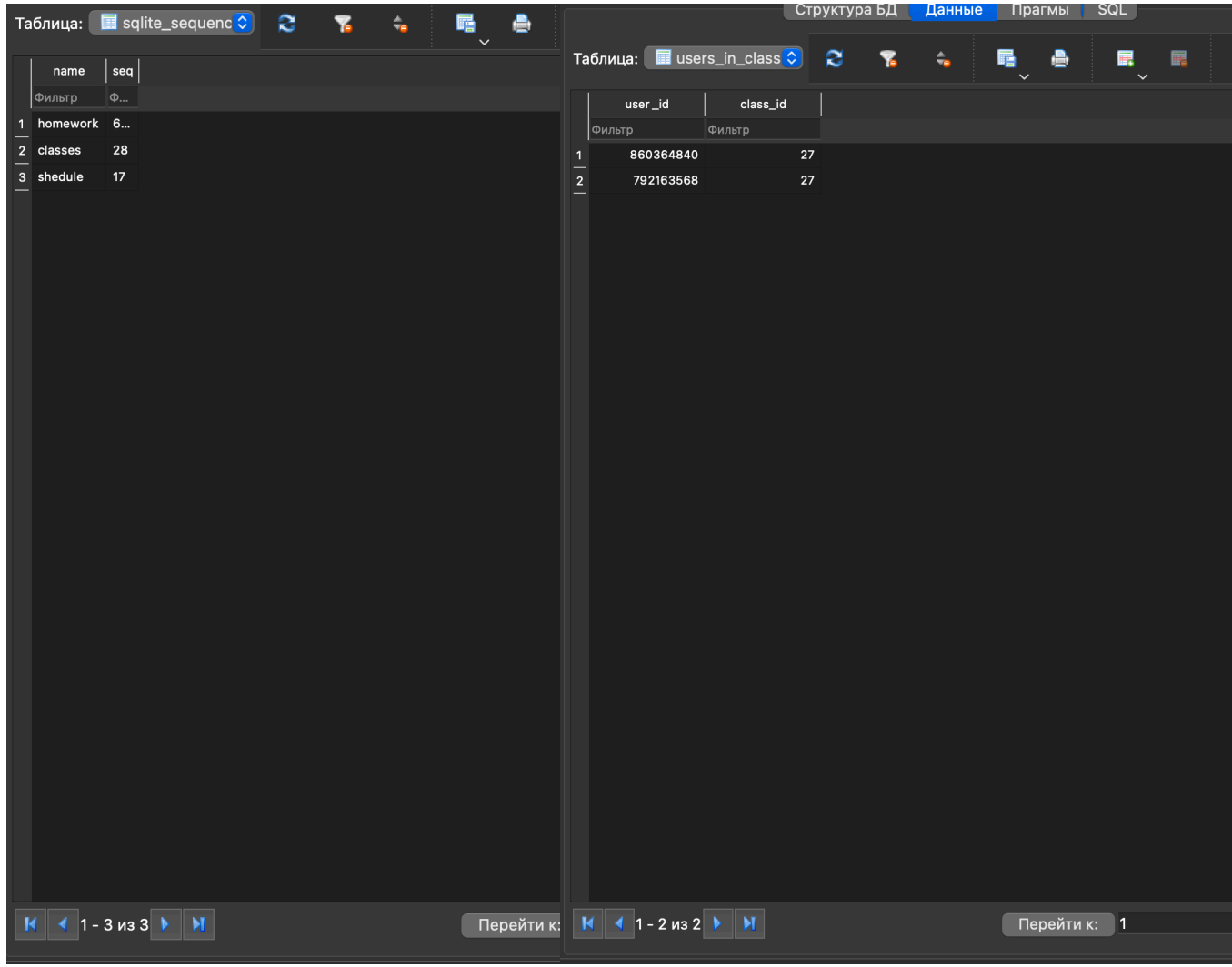
Идея и суть проекта:

- Разработка бота с помощью Telegram API
- Создание электронного дневника для школьников

Создание БД

Админы, классы, домашние задания, Расписания, ученики

- Внесение информации об админах
- Создание классов
- Хранение домашних заданий по дням
- Хранение расписания на каждый день
- Информация о каждом ученике



Меню, Функционал

Простое и доступное меню, необходимый функционал дневника

- Стартовое окно с кнопками
- Вызов определенной функции по нажатию на соответствующую кнопку
- Хранение домашних заданий по дням
- Хранение расписания на каждый день
- Информация о каждом ученике

```python
# команда /start
@bot.message_handler(commands=['start', 'help'])
def start_message(message):
    # получаем имя user и здороваемся с ним
    user_first_name = str(message.chat.first_name)
    # создаём список с нужными для нас кнопками
    buttons = ['👩Создать класс', '🧛Найти класс', '❓Связаться с разработчиками', '🎓Ваши классы', '🆔Получить id']
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
    # добавление кнопок из списка на главный экран
    for button in buttons:
        markup.add(button)
    # отправляем сообщения пользователю, который написал команду /start
    bot.send_message(message.chat.id, f'Привет, {user_first_name}!\nТебя приветствует SchoolBot,'
                                      f' пользуйся ботом с помощью команд снизу👇.', reply_markup=markup)
```

Работа с БД

Внесение, получение информации

- В зависимости от вызванной команды мы получаем информацию из БД или же вносим информацию в БД

```python
# добавляем админа
def add_admin(self, key):
    # проверка на существование класса
    result_of_execute = self.cursor.execute(f'SELECT id FROM classes WHERE key'
                                            f' = {key}').fetchall()
    if not result_of_execute:
        return False
    # внесение в БД информации о админе
    sqlite_insert_query = f"""INSERT INTO admins (class_id, admin)  VALUES
      ({result_of_execute[0][0]}, {self.user_id})"""
    self.cursor.execute(sqlite_insert_query)
    self.con.commit()
```

Команды пользователя

Работа с пользователем

- Каждая кнопка отвечает за свой функционал, тем самым пользователь работает с ботом так, как ему нужно
- Бот предусматривает два типа пользователей : учитель (админ) и ученик
- Учитель может создавать классы, ученики в них заходить
- Функционал дневника зависит от того, является пользователь учителем или учеником

```python
# создание класса по вызову от нажатой кнопки
def create_class(message):
    global ACTIVE_CLASS
    try:
        # Выход из функции при необходимости вернуться на главную
        if message.text == '✅Назад в главную':
            return start_message(message)
        elif message.text == '/start':
            return start_message(message)
        # создание уникального ключа
        creating_key = True
        # начальное значение ключа
        key = ''
        # подключение к БД
        sqlighter = SQLighter(message.from_user.id)
        # генерация случайного ключа
        while creating_key:
            # добавление случайных значений в ключ
            key = ''.join(random.choice(string.digits) for _ in range(6))
            # проверка уникальности ключа
            if key[0] != '0':
                if not sqlighter.add_class(key, message.text):
                    creating_key = False
        # присваиваем постоянной переменной ACTIVE_CLASS значение ключа
        ACTIVE_CLASS = key
        # добавление админа в класс по его ключу
        sqlighter.add_admin(key)
        # добавление пользователя в класс по его ключу
        sqlighter.add_user_to_class(key)
        markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
        # создание списка необходимых кнопок
        buttons = ['📓Расписание', '📒ДЗ', '⚙Настройки',
                   '📝Объявление', '✅Назад в главную']
```

ЗАКЛЮЧЕНИЕ

Итоги и доработка

Итоги

- В итоге всей работы мы создали школьного бота в Телеграм, который может заменить обычный дневник и дать конкуренцию сайту dnevnik.ru.
- В ходе работы были изучены возможности API telegram, научились пользоваться библиотекой telebot.

Возможности по доработке

- Оптимизация под большое количество пользователей
- Добавление нового функционала, такого как выставление оценок