Белорусский Государственный Университет Информатики и Радиотехники
Факультет Компьютерных Систем и Сетей
Кафедра Информатики

КУРСОВАЯ РАБОТА

по предмету
«Объектно-ориентированное программирование»
Тема: Разработка Бота для нахождения арендной торговой площади

Выполнил:
студент гр.953505
Ивашкевич В.А.
_____

Проверил:
доцент
Летохо А.С.
_____

г. Минск, 2021 год.

Содержание

Введение

Современный мир немыслим без интернет-технологий, которые постоянно развиваются. То, что вчера казалось фантастикой, сегодня становится реальностью. Особенно бурно развиваются новые системы искусственного интеллекта. Виртуальный собеседник перестает быть живым человеком, на его смену приходят мощные программы. Их функция не только развлекать, но и быть надежным помощником в повседневной работе. Таковы чат-боты – самая прогрессивная и перспективная технология в мире интернета.

ГЛАВА 1

1.1 Что такое чат-боты

Если говорить простым языком, то чат-боты – это виртуальные собеседники, программы имитирующие живого человека. В основу работы положен алгоритм искусственного интеллекта. Взаимодействие обычно проходит через интернет-чат.

Самые часто встречающиеся – это электронные менеджеры, которые консультируют пользователя при посещении сайта. Однако, функции чат-ботов постоянно расширяются. Сейчас такие программы уже напоминают личных секретарей, в круг их обязанностей входят информирование о погоде, о скидках, заказ билетов и многое другое. Существует мнение, что чат-боты в скором времени вытеснят социальные сети и заменят целый ряд сайтов. На их стороне удобство и эффективность.

1.2 Как создавались чат-боты

Первопроходцем в этом деле стал американец Джозеф Вейценбаум, профессор Массачусетского технологического университета. В далеком 1966 году он написал первого бота, – «Элиза». Это был виртуальный собеседник, который сразу задал высокую планку. Далеко не все пользователи смогли определить своего визави как компьютер. С тех пор идею подхватили, значительно усовершенствовав первые разработки. Особенно чат-ботами заинтересовались при появлении интернета в конце 90-х. Уже тогда многие понимали, что электронный собеседник гораздо удобнее и надежнее. С конца 90-х американцы, японцы и китайцы соревновались за лучший ИИ (искусственный интеллект), и теперь мы имеем массу вариаций чат-ботов, которые постоянно учатся.

1.3 На чем основана работа чат-бота

Мало кто из пользователей готов создать своего чат-бота с нуля. Это потребует некоторых знаний и времени, тем более, что доступно масса

сервисов, которые все сделают за тебя. Достаточно знать, что чат-бот может быть создан на любом языке программирования, который имеет функцию web API. Чаще всего встречаются Node.js и PHP, а также многие другие библиотеки, поддерживающие Java или Python. Все известные мессенджеры всегда предоставляют подробные руководства, как связать своего чат-бота с их платформой.

Распространение получили два вида ботов: первые функционируют на основе набора правил, а второй обладает машинным обучением. Это означает, что первый тип реагирует исключительно на предельно конкретные команды, отхождения от которых делают чат-боты бесполезными. Такой бот будет действовать только в рамках заданных алгоритмов.

Более продвинутый вариант обладает искусственным интеллектом, т.е. машинным обучением. Эти боты уже понимают человеческий язык, а не набор команд. Возможности общения значительно шире. Кроме того чат-боты обучаются после каждого сеанса общения.

1.4 Современные возможности чат-ботов

Благодаря своим качествам приобрели большое распространение. В список возможностей входят консультирование по вопросам:

- медицины;
- закона и права;
- страхования;
- покупок и продаж;
- инвестирования и других областей.

Фактически современный бот способен ответить на любой вопрос, достаточно только загрузить в него информацию. Главное, что такая программа работает быстро и почти всегда безошибочно, полностью заменяя человека.

Одной из популярных функций является органайзер, когда чат-бот самостоятельно составляет расписание, анализируя полученный код и предоставляя несколько вариантов распределения времени. Это очень удобно, когда у человека плотный график. Фактически бот выполняет роль секретаря, помогая экономить время и деньги.

Чат-боты заняли место помощников в поисковиках и месседжерах. Так «Яндекс» и уже давно активно предлагают своим пользователям вспомогательные программы. В круг их обязанностей входят как простой поиск, так и аналитика.

В развлекательном секторе чат-боты используются как собеседник или обучающая программа. Особенно интересно для детей.

В онлайн играх не редко применяются боты, которые автоматизируют процесс игры, когда фактически за юзера играет компьютер. Игрок лишь отдает основные команды. Это сильно экономит время.

Более серьезное применение – это биржевые чат-боты. Их задача отслеживать и анализировать всю информацию по торговой площадке и давать рекомендации брокеру. Конечно, такой бот не заменит человека, но значительно облегчит его труд. Доля совершаемых ботами сделок достигает 30%, что не мало.

Современные возможности чат-ботов постоянно совершенствуются и расширяются. Сейчас их можно встретить и в социальных сетях и в серьезном бизнесе. Искусственный интеллект не ошибается, при этом значительно экономит время. В эти технологии сейчас вкладываются большие деньги, и круг распространения очень широк.

1.5 Виды и функции чат-ботов

Чат-боты не всегда безопасны. Так их давно уже адаптируют под криминальные нужды. Не редки случаи ботов вымогателей или выясняющих пароли и логины пользователей.

Выделяют несколько видов чат-ботов:

- Консультанты – от простых менеджеров в интернет-магазинах до медицинских и юридических услуг. Их обязанности – общаться с потенциальным и реальным клиентом, отвечая на всевозможные вопросы.
- Помощники – можно встретить в мессенджерах и поисковиках. Помогают при поиске информации, также проводят ее первичный анализ.
- Развлечения – часто виртуальный собеседник, отвечающий на вопросы. Программы обучаются на оставленных сообщениях, и со временем могут заменить реального собеседника.
- Бизнес чат-боты – задача этих программ оптимизировать работу, сделав ее более эффективной. Не работают за тебя, но делают ее легче. Бизнес чат-боты – задача этих программ оптимизировать работу, сделав ее более эффективной. Большой популярностью пользуются у брокеров и трейдеров, когда нужно срочно узнать состояние биржи. Также отлично зарекомендовали себя для сбора статистических данных о той или иной онлайн-компании. Не редко используются как обычный ежедневник.

Функция чат-ботов лежит в плоскости оперативных услуг, когда надо быстро и в любой момент ответить на вопрос или выполнить анализ ситуации. Позволяет экономить на веб-персонале, делая работу автоматизированной.

1.6 Чат-бот в социальных сетях

Социальные сети по достоинству оценили возможности чат-ботов. Известно, что в популярной сети ВК почти половину аккаунтов занимают боты и виртуалы. Однако в соц. сетях чат-боты часто несут полезные функции, как то: узнать курс валют, последние новости, выбрать билеты или любой другой товар, составить гороскоп, перевести слово. Часто эффективно заменяют

собой мобильные приложения, поскольку легче устанавливаются и стоят дешевле.

В негативном свете проявляют себя как бесконечные генераторы спама. Для борьбы с ними используются контрпрограммы.

При правильном подходе при отсутствии фантазии чат-бот поможет завести новых друзей, взяв на себя функцию переговорщика. Сейчас уже есть очень «умные» боты, способные максимально мимикрировать под человека.

1.7 Чат-бот в бизнесе

Самый простой пример чат-ботов в бизнесе, это программы Битрикс24 или SpyCat 2.0. Помогают оптимизировать работу и увеличить КПД. Благодаря чат-ботам достигается автоматизация рутинных процессов, за счет чего экономится время и деньги. Также это удобно и для клиента, который получает информацию напрямую.

Другая важная функция чат-ботов в бизнесе – это аналитика. Такие программы на финансовом рынке пользуются большой популярностью за счет быстроты реагирования. Иногда чат-боты используются в не совсем законной деятельности, когда нужно создать видимость поддержки некоего события.

Распространенные виды бизнес чат-ботов: еженедельник с функцией органайзера, личный секретарь, счетчик обмена валют с постоянным обновлением, программы анализа состояния финансового рынка.

1.8 Чат-боты в других сферах

Особенно перспективным видится будущее чат-ботов как обучающих программ для детей. Они не только способны отвечать на многие вопросы, но и предоставлять учебный материал в игровом виде. Также интересны чат-боты в виде электронных энциклопедий. Программа находит по ключевому слову необходимую информацию и комментирует ее.

Чат-боты популярны в развлекательном секторе. Виртуальный собеседник не только отвечает на вопросы, но и обучается в процессе.

1.9 Преимущества и недостатки чат-бота

Чат-боты прочно завоевали свою нишу, и на это есть несколько причин:

- как компьютер, они прекрасно справляются с вычислительными операциями и проводят анализ баз данных за секунды, выдавая оптимальное решение.
- автоматизирует рутинную деятельность, позволяя экономить на персонале.
- как помощник бот мало чем отличается от человека, при этом работает в разы быстрее.
- чат-бот быстро ищет. Это очень важно при работе с большими объемами данных, например в юриспруденции.
- чат-боты быстро совершенствуются и в некоторых областях способны полностью заменить человека.

К недостаткам относятся:

- много функций в ущерб эффективности. Стремясь создать универсального чат-бота, программисты часто недорабатывают отдельные компоненты, отчего вся работа происходит не совсем корректно.
- простейшие алгоритмы. Чтобы создать мощного бота, надо обладать большой технологической базой. Это недоступно простому пользователю.
- несовершенный интерфейс. Обычно с чат-ботом приходится общаться путем текстового набора. Искусственный интеллект далеко не всегда понимает, что у него спрашивается и тем более не может реагировать на эмоции.

Несмотря на недостатки, есть области, когда чат-бот хорошо себя зарекомендовал. Тем более, что они постоянно модернизируются.

ГЛАВА 2 Логическое проектирование

2.1 Основные этапы:

1.Выберите специализацию

Не надо пытаться угодить всем и собирать много разных функций, ничего хорошего из этого не выйдет. Определите конкретные функционал, тематику и аудиторию — сконцентрируйтесь на них.

2.Создайте индивидуальность

Аватарка, словарный запас, набор эмотиконов, тон общения и вообще любые составляющие интерфейса вашего бота ориентируйте на ту аудиторию и задачу, которую вы выбрали. При этом пользователь должен знать, что он общается с роботом — не пытайтесь обманывать и выдавать бота за человека. Но для комфортного взаимодействия бот всё же должен имитировать некоторые человеческие качества.

3.Пропишите список обязательных функций

Какая информация скорее всего понадобится любому человеку, который собирается пользоваться вашим ботом? Какие основные действия бот будет совершать? Для вывода справки рекомендуется использовать традиционные для платформы команды — например, `/help` в Telegram.

4.Обеспечьте безопасность

Ваш бот должен быть защищён от вероятных атак, тем более, если он уже интегрирован с CRM, внутренними серверами или другими важными системами.

5.Продумайте логику общения

Создайте дерево сценариев
Вам нужно учесть все возможные варианты развития диалога с пользователем: от самых неудачных до успешных.

6.Избегайте тупиков
Максимально конкретизируйте взаимодействие: беседа должна быть последовательной, сообщения от чат-бота — лаконичными. Не надо ставить пользователя в ситуации, в которых он может «потеряться»: например, даже простой вопрос, на который возможны два ответа — «да» или «нет», лучше оформить в виде кнопок.

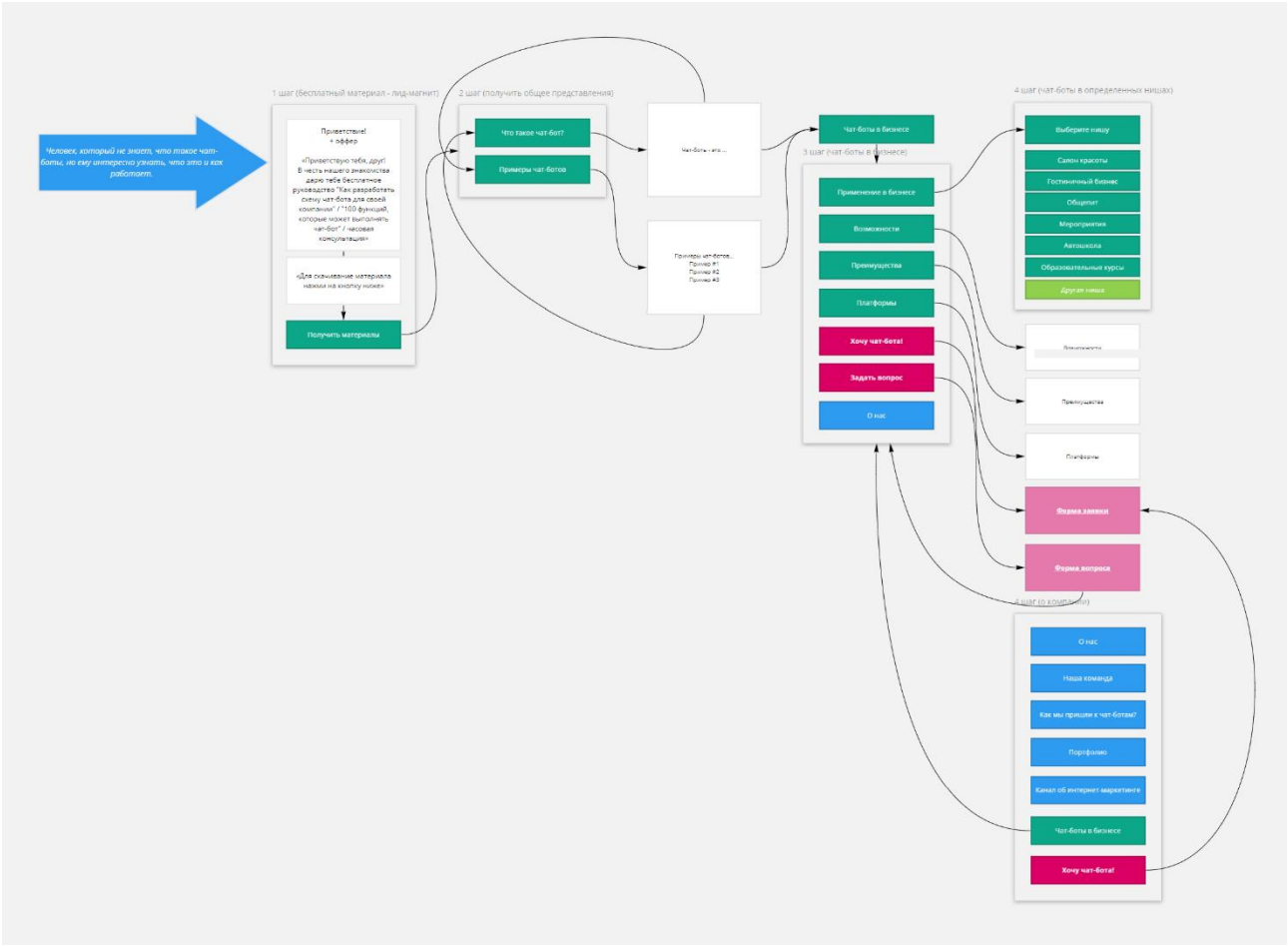
7.Добавьте кнопки
Сделать клик — быстрее и удобнее, чем набирать текст, особенно на бегу. К тому же кнопка подскажет пользователю, какого рода запросы можно отправлять боту.

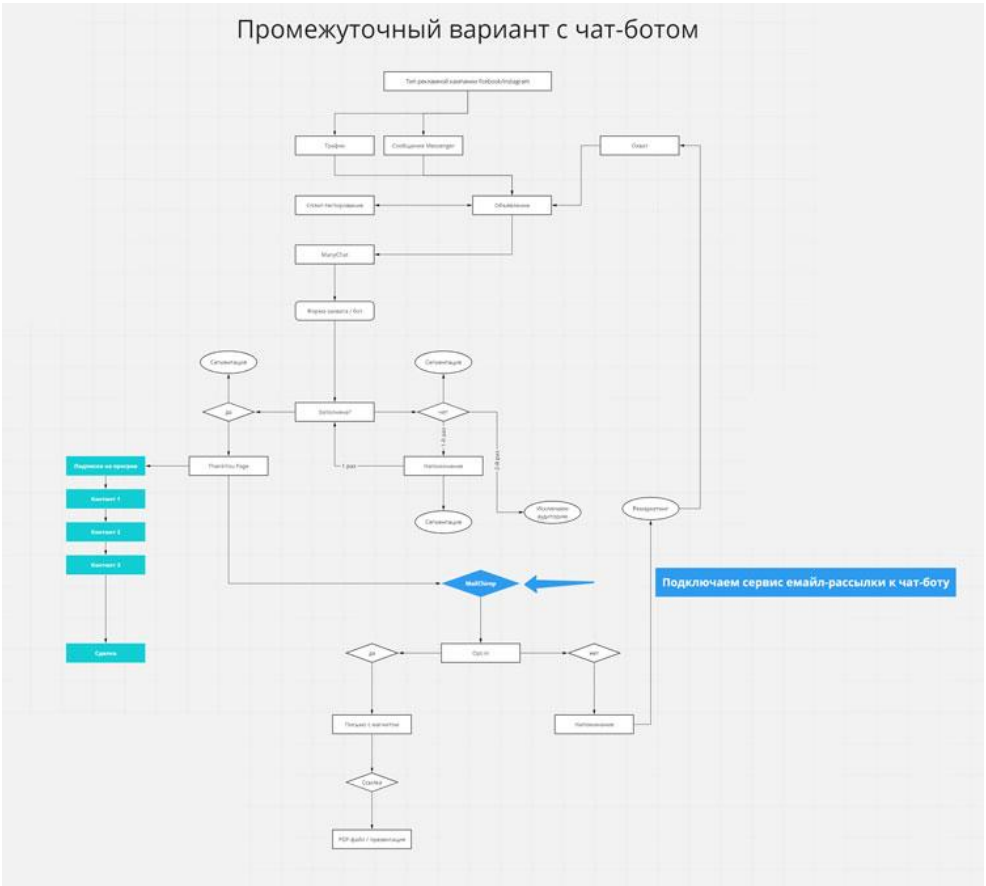
8.Используйте изображения
Сейчас добавить изображение можно на всех платформах, в том числе в СМС. Картинка привлекает внимание к сообщению и в связке с подписью эффективнее, чем просто текст. Но не злоупотребляйте.

9.Сделайте сообщения об ошибках полезными
После фразы о том, что у бота что-то не получилось, добавьте какое-то предложение пользователю, хотя бы «вернуться в главное меню». Не оставляйте тупиков, об этом говорилось выше.

10.Укажите канал связи с реальным человеком
Всегда будут запросы, с которыми бот справиться не может, поэтому нужно убедиться, что хотя бы часть из них (которую вы определяете сами), адресуется человеку. К тому же у пользователя должна быть под рукой команда, которая выражает намерение «не хочу более общаться с ботом, дайте мне связаться с человеком». Но продумывайте такие вещи тщательно

2.2 Примеры схем логического дерева


Человек, который не знает, что такое чат-боты, но ему интересно узнать, что это и как работает.
1 шаг (бесплатный материал - лид-магнит)
Приветствие!
+ оффер
«Приветствую тебя, друг!
В честь нашего знакомства дарю тебе бесплатное руководство "Как разработать схему чат-бота для своей компании" / "100 функций, которые может выполнять чат-бот" / часовая консультация»
«Для скачивание материала нажми на кнопку ниже»
Получить материалы
2 шаг (получить общее представление)
Что такое чат-бот?
Примеры чат-ботов
Чат-боты - это ...
Примеры чат-ботов...
Пример #1
Пример #2
Пример #3
Чат-боты в бизнесе
3 шаг (чат-боты в бизнесе)
Применение в бизнесе
Возможности
Преимущества
Платформы
Хочу чат-бота!
Задать вопрос
О нас
4 шаг (чат-боты в определенных нишах)
Выберите нишу
Салон красоты
Гостиничный бизнес
Общепит
Мероприятия
Автошкола
Образовательные курсы
Другая ниша
Преимущества
Платформы
Форма заявки
Форма вопроса
4 шаг (о компании)
О нас
Наша команда
Как мы пришли к чат-ботам?
Портфолио
Канал об интернет маркетинге
Чат-боты в бизнесе
Хочу чат-бота!


Промежуточный вариант с чат-ботом
Подключаем сервис емайл-рассылки к чат-боту

Глава 3 Написание кода

Часть 1: Регистрация бота

Самая простая и описанная часть. Очень коротко: нужно найти бота *@BotFather*, написать ему */start*, или */newbot*, заполнить поля, которые он спросит (название бота и его короткое имя), и получить сообщение с токеном бота и ссылкой на документацию. Токен нужно сохранить, желательно надёжно, так как это единственный ключ для авторизации бота и взаимодействия с ним.

Часть 2: Подготовка к написанию кода

Как уже было сказано в заголовке, писать бота мы будем на Python'е. В данной статье будет описана работа с библиотекой PyTelegramBotAPI (Telebot). Если у вас не установлен Python, то сперва нужно сделать это: в терминале Linux нужно ввести

```
sudo apt-get install python python-pip
```

Если же вы пользуетесь Windows, то нужно скачать Python с официального сайта .

После, в терминале Linux, или командной строке Windows вводим

```
pip install pytelegrambotapi
```

Теперь все готово для написания кода.

Часть 3: Получаем сообщения и говорим «Привет»

Небольшое отступление. Телеграмм умеет сообщать боту о действиях пользователя двумя способами: через ответ на запрос сервера (Long Poll), и через Webhook, когда сервер Телеграмма сам присылает сообщение о том, что кто-то написал боту. Второй способ явно выглядит лучше, но требует выделенного IP-адреса, и установленного SSL на сервере. В этой статье я хочу рассказать о написании бота, а не настройке сервера, поэтому пользоваться мы будем Long Poll'ом.

Открывайте ваш любимый текстовый редактор, и давайте писать код бота!

Первое, что нужно сделать это импортировать нашу библиотеку и подключить токен бота:

```
import telebot;
```

```
bot = telebot.TeleBot('%ваш токен%');
```

Теперь объявим метод для получения текстовых сообщений:

```
@bot.message_handler(content_types=['text'])
```

```
def get_text_messages(message):
```

В этом участке кода мы объявили слушателя для текстовых сообщений и метод их обработки. Поле content_types может принимать разные значения, и не только одно, например

```
@bot.message_handler(content_types=['text', 'document', 'audio'])
```

Теперь добавим в наш метод немного функционала: если пользователь напишет нам «Привет», то скажем ему «Привет, чем я могу помочь?», а если нам напишут команду «/help», то скажем пользователю написать «Привет»:

```
if message.text == "Привет":
```

```
    bot.send_message(message.from_user.id, "Привет, чем я могу тебе
```

```
помочь?")
```

```
elif message.text == "/help":
    bot.send_message(message.from_user.id, "Напиши привет")
else:
    bot.send_message(message.from_user.id, "Я тебя не понимаю. Напиш
и /help.")
```

Данный участок кода не требует комментариев, как мне кажется. Теперь нужно добавить в наш код только одну строчку (вне всех методов).

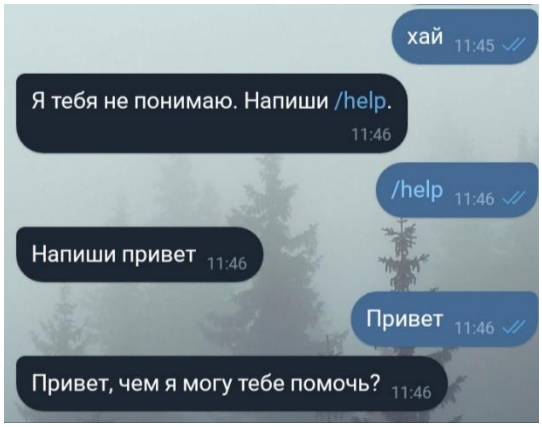
```
bot.polling(none_stop=True, interval=0)
```

Теперь наш бот будет постоянно спрашивать у сервера Телеграмма «Мне кто-нибудь написал?», и если мы напишем нашему боту, то Телеграмм передаст ему наше сообщение. Сохраняем весь файл, и пишем в консоли

```
python bot.py
```

Где bot.py – имя нашего файла.

Теперь можно написать боту и посмотреть на результат:



Часть 4: Кнопки и ветки сообщений:

Markup- место расположение наших кнопок , данный код создает 3 кнопки на выбор

1.Все варианты – выдаст все варианты имеющиеся в базе

2.По площади – клиент сможет выбрать интересующую его площадь из предложенных вариантов (100<) или (<100)

3.По цене – клиент сможет выбрать интересующий его ценовой диапазон (<300$) , (300-400$), (400-500$), (500-600$)

```
47
48
49  @bot.message_handler(commands=['start'])
50  def welcome(message):
51
52      # keyboard
53      markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
54      item1 = types.KeyboardButton("Все варианты")
55      item2 = types.KeyboardButton("По площади")
56      item3 = types.KeyboardButton("По цене")
57
58      markup.add(item1, item2,item3)
59
60      bot.send_message(message.chat.id, "Добро пожаловать, {0.
61          parse_mode='html', reply_markup=markup)
62
```

5.Хранение данных

Все объекты у нас будут однотипны, поэтому для упрощенного варианта их хранения – создадим класс Place.

```
class Place(object):
    emp_count = 0

    def __init__(self, adres, square, price, number_own,picture):
        self.adres = adres
        self.square = square
        self.price = price
        self.number_own = number_own
        self.picture =picture

        Place.emp_count += 1

    def display_count(self):
        return Place.emp_count

    def get_square(self):
        return self.square

    def get_price(self):
        return self.price

    def get_picture(self):
        return self.picture

    def display_place(self):
        return f'Адрес: {self.adres}.\n Площадь: {self.square} метров квадратных. \n Цена за месяц: {self
```

Каждый объект будет обладать следующими свойствами:
1.Адресс
2.Площадь
3.Цена
4.Номер владельца
5.Фотография

Обрабатываем вариант нажатия кнопки “Все варианты”. Нам нужно вывести все имеющиеся варианты, для этого воспользуемся обычным перебором, при помощи цикла for.

```
@bot.message_handler(content_types=['text'])
def lalala(message):
    if message.chat.type == 'private':
        if message.text == 'Все варианты':
            for i in range(0,len(mas)):
                p = mas[i].display_place()
                pic=mas[i].get_picture()
                bot.send_message(message.chat.id, f" {p}")
                img = open(f'{pic}', 'rb')
                bot.send_photo(message.chat.id, img)
```

В варианте “По площади ” будет немного сложнее, так как после, мы предложим варианты при помощи создания подстроковых кнопок

```
elif message.text == 'По площади':

    markup = types.InlineKeyboardMarkup(row_width=2)
    item1 = types.InlineKeyboardButton(">=100", callback_data='good')
    item2 = types.InlineKeyboardButton("<100", callback_data='bad')

    markup.add(item1, item2)

    bot.send_message(message.chat.id, 'какая площадь вас интересует?(в м^2)', reply_markup=markup)
```

Данные кнопки будут отправлять сигнал для последующих действий (сортировка массива с вариантами)

Затем обрабатываем принятие сигнала, первый вариант

```
@bot.callback_query_handler(func=lambda call: True)
def callback_inline(call):
    try:
        if call.message:
            if call.data == 'good':
                count=0
                bot.send_message(call.message.chat.id, 'Вот варианты:\n')
                for i in range(0,len(mas)):
                    if(int(mas[i].get_square())>=100):
                        count+=1
                        p = mas[i].display_place()
                        bot.send_message(call.message.chat.id, f" {p}")
                        pic=mas[i].get_picture()
                        img = open(f'{pic}', 'rb')
                        bot.send_photo(call.message.chat.id, img)

                if(count==0):
                    bot.send_message(call.message.chat.id,"К сожалению вариантов не нашлось , попробуйте
```

Затем второй

```
    elif call.data == 'bad':
        bot.send_message(call.message.chat.id, 'Можем предложить такие варианты:')
        count2=0
        for i in range(0,len(mas)):
            if(int(mas[i].get_square())<100):
                count2+=1
                p = mas[i].display_place()
                bot.send_message(call.message.chat.id, f" {p}")
                pic=mas[i].get_picture()
                img = open(f'{pic}', 'rb')
                bot.send_photo(call.message.chat.id, img)

        if(count2==0):
            bot.send_message(call.message.chat.id,"К сожалению вариантов не нашлось , попробуйте
```

После написания данного кода, мы сможем получать нужные нам элементы массива

Теперь обработаем вариант нажатия “По цене”

Для этого используем тот же метод (сигнала)

```
elif message.text == 'По цене':

    markup = types.InlineKeyboardMarkup(row_width=4)
    item1 = types.InlineKeyboardButton("<300$", callback_data='small')
    item2 = types.InlineKeyboardButton("300-400$", callback_data='middle')
    item3 = types.InlineKeyboardButton("400-500$", callback_data='middle2')
    item4 = types.InlineKeyboardButton("500-600$", callback_data='large')

    markup.add(item1, item2,item3,item4)

    bot.send_message(message.chat.id, 'какая цена вас интересует?', reply_markup=markup)
```

Обрабатываем диапазон (<300$)

```
elif call.message:
        if call.data == 'small':
            count3=0
            bot.send_message(call.message.chat.id, 'Вот варианты:\n')
            for i in range(0,len(mas)):
                if(int(mas[i].get_price())<300):
                    count3+=1
                    p = mas[i].display_place()
                    bot.send_message(call.message.chat.id, f" {p}")
                    pic=mas[i].get_picture()
                    img = open(f'{pic}', 'rb')
                    bot.send_photo(call.message.chat.id, img)

            if(count3==0):
                bot.send_message(call.message.chat.id,"К сожалению вариантов не нашлось , поп
```

Обрабатываем диапазон (300-400$)

```
elif call.data == 'middle':
        bot.send_message(call.message.chat.id, 'Можем предложить такие варианты:')
        count4=0
        for i in range(0,len(mas)):
            if((int(mas[i].get_price())>=300) and (int(mas[i].get_price()<400))):
                count4+=1
                p = mas[i].display_place()
                bot.send_message(call.message.chat.id, f" {p}")
                pic=mas[i].get_picture()
                img = open(f'{pic}', 'rb')
                bot.send_photo(call.message.chat.id, img)

        if(count4==0):
            bot.send_message(call.message.chat.id,"К сожалению вариантов не нашлось , попробу
```

Обрабатываем диапазон(400-500$)

```
elif call.data == 'middle2':
        bot.send_message(call.message.chat.id, 'Можем предложить такие варианты:')
        count5=0
        for i in range(0,len(mas)):
            if(int(mas[i].get_price())>=400 and int(mas[i].get_price()<500)):
                    count5+=1
                    p = mas[i].display_place()
                    bot.send_message(call.message.chat.id, f" {p}")
                    pic=mas[i].get_picture()
                    img = open(f'{pic}', 'rb')
                    bot.send_photo(call.message.chat.id, img)

        if(count5==0):
            bot.send_message(call.message.chat.id,"К сожалению вариантов не нашлось , попробу
```

И последний (500-600$)

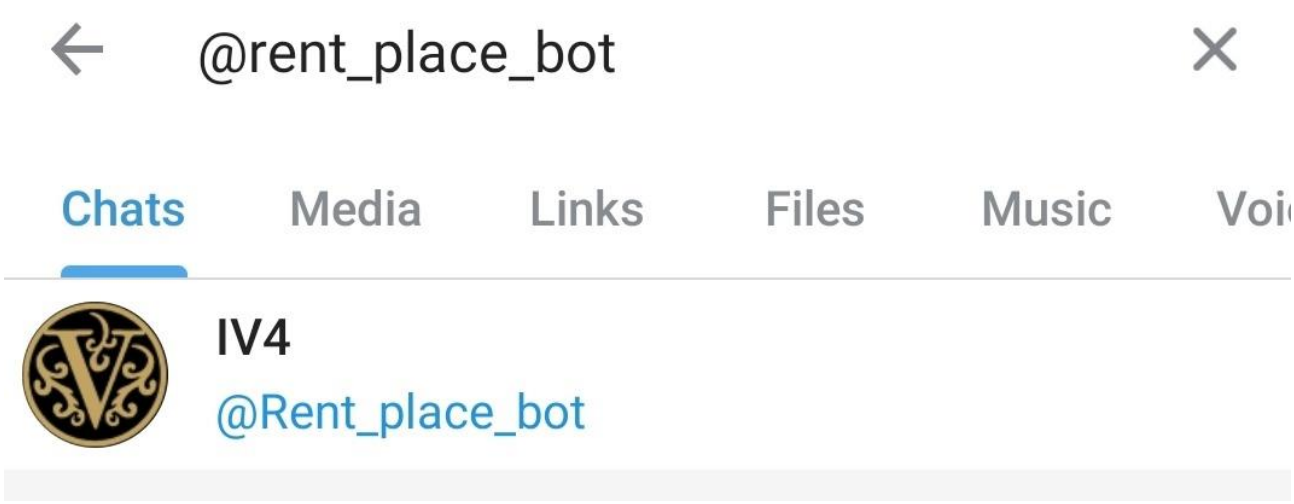
```
elif call.data == 'large':
        bot.send_message(call.message.chat.id, 'Можем предложить такие варианты:')
        count6=0
        for i in range(0,len(mas)):
                if(int(mas[i].get_price())>=500 and int(mas[i].get_price()<600)):
                    count6+=1
                    p = mas[i].display_place()
                    bot.send_message(call.message.chat.id, f" {p}")
                    pic=mas[i].get_picture()
                    img = open(f'{pic}', 'rb')
                    bot.send_photo(call.message.chat.id, img)

        if(count6==0):
            bot.send_message(call.message.chat.id,"К сожалению вариантов не нашлось , попробуй
```

Так же нам нужна проверка на неизвестные сообщения

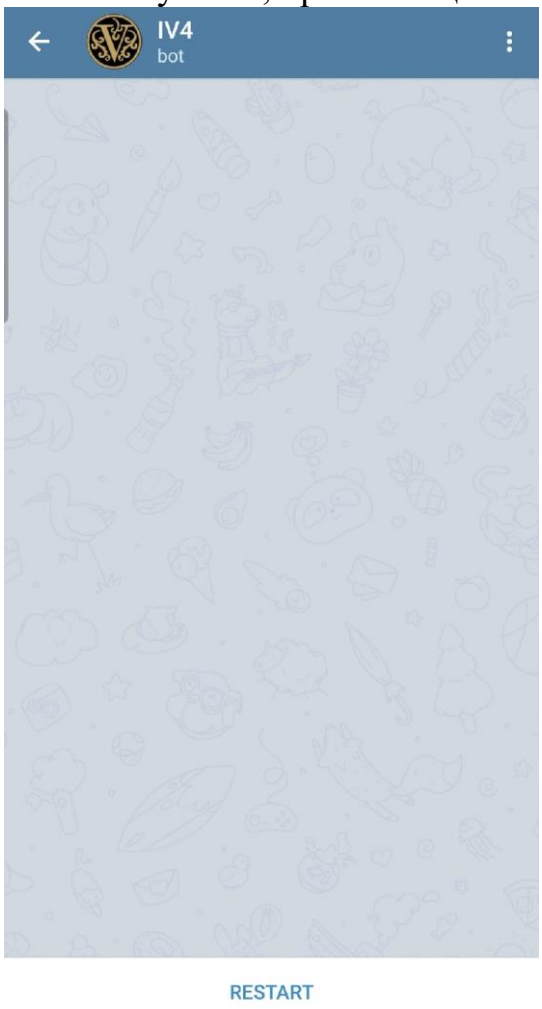
```
else:
    bot.send_message(message.chat.id, 'Я не знаю что ответить 😢')
```

Глава 4 Пользование чат-ботом

Для того, чтобы начать пользоваться чат-ботом, надо найти его в общем поиске

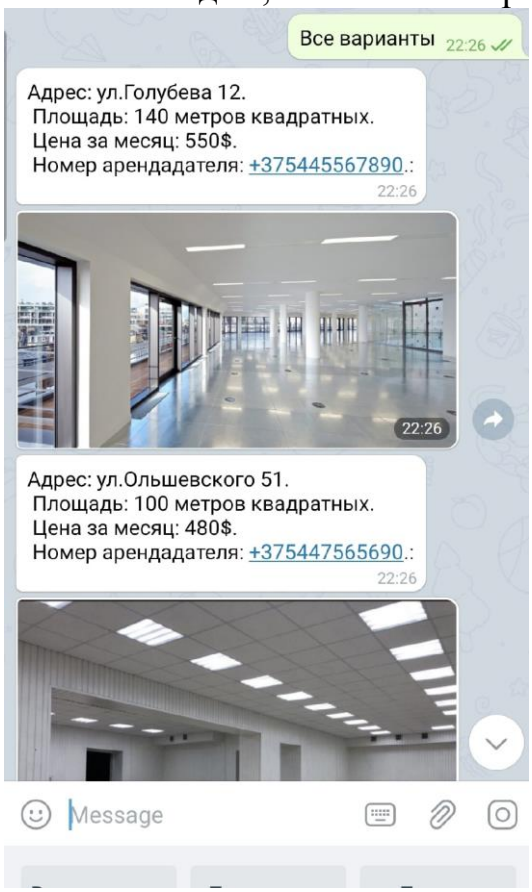


Затем запустить, при помощи кнопки Start



После чего, у вас будут подсказки в виде кнопок

И как мы видим, наш чат-бот прекрасно работает



Выводы

Согласно проведенному Facebook исследованию, в зависимости от региона, от 66% до 76% пользователей относятся лучше к бренду, с которым могут вести онлайн чат. Цифры исследования очень важны для понимания феномена чат-ботов. По сути, боты являются ответом рынка на уже *сформированный* общественный запрос.

Другое исследование провела среди своих потребителей в США и Германии провайдерская компания «Ovum» в 2016 году. Целью было определение основных причины выбора именно онлайн чата потребителями.

Лидирует желание разрешить свой вопрос как можно скорее и нежелание общаться более чем через один канал связи.

Сейчас мгновенная реакция на запросы клиентов — не преимуществом, а необходимостью. Согласно опросам InsideSales и Harvard Business Review, пятиминутная задержка ответа от компании может привести к тому, что каждый десятый клиент уйдет. Ответ через десять минут уменьшит ваши шансы получить покупателя на 400%.
Чат-боты действительно самый простой способ решить эту проблему. Телефонные звонки, электронная почта и даже красивые веб-формы не предлагают мгновенную реакцию, которую дает большинство современных чат-ботов. Онлайн-чаты с сотрудниками можно рассматривать как коммуникационный канал веб-сайта, но в конечном итоге и чат-боты могут быть активными через него. Любой бизнес, который использует онлайн-чаты *вместе* с чат-ботами, решит проблемы быстрее.

Еще одной причиной экспансии чат-ботов является так называемая «усталость от приложений». Людей раздражает необходимость устанавливать специальные приложения на свои мобильные устройства — это отнимает время и силы. К тому же, нередко они устанавливаются для одноразового использования. А создавать, продвигать и вводить в действие новые приложения, которые в итоге не будут использоваться — невыгодно. Будучи способными функционировать на разных планах и через разные каналы, чат-боты решают и эту проблему.

Список используемой литературы

«Изучаем Python», Марк Лутц

«Высокопроизводительный Python: практическое пособие для людей», Миша Горелик, Ян Освальд

«Python. Разработка на основе тестирования», Гарри Персиваль

Приложение (код)

```
from logging import StringTemplateStyle
import telebot
import os

import random

from telebot import types

class Place(object):
    emp_count = 0

    def __init__(self, adres, square, price, number_own,picture):
        self.adres = adres
        self.square = square
        self.price = price
        self.number_own = number_own
        self.picture =picture

        Place.emp_count += 1

    def display_count(self):
        return Place.emp_count

    def get_square(self):
        return self.square

    def get_price(self):
        return self.price

    def get_picture(self):
        return self.picture

    def display_place(self):
        return f'Адрес: {self.adres}.\n Площадь: {self.square} метров квадратных. \
n Цена за месяц: {self.price}$. \n Номер арендадателя: {self.number_own}.: \n '
place1=Place("ул.Голубева 12",'140','550','+375445567890','E:\_bot_curs\picture1.jp
g')
place2=Place("ул.Ольшевского 51",'100','480','+375447565690','E:\_bot_curs\picture2
.jpg')
place3=Place("ул.Тараканова 4А",'89','300','+375290986745','E:\_bot_curs\picture3.j
pg')
place4=Place("пр.Дзержинского 15",'70','250','+375293476745','E:\_bot_curs\picture4
.jpg')
place5=Place("пр.Правды 7",'110','430','+375290980005','E:\_bot_curs\picture5.jpg')

mas = [place1,place2,place3,place4,place5]


#-------------------------------------------------------------------------------
-------------------------
 #BOT
bot = telebot.TeleBot('1715199163:AAG_nJC8vpbL1ignSxUL8oZ-kHxhbaVCffA')
```

```
@bot.message_handler(commands=['start'])
def welcome(message):

    # keyboard
    markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
    item1 = types.KeyboardButton("Все варианты")
    item2 = types.KeyboardButton("По площади")
    item3 = types.KeyboardButton("По цене")

    markup.add(item1, item2,item3)

    bot.send_message(message.chat.id, "Добро пожаловать, {0.first_name}!\nЯ -
 <b>{1.first_name}</b>, бот созданный чтобы помочь тебе найти варианты аренды площа
ди".format(message.from_user, bot.get_me()),
        parse_mode='html', reply_markup=markup)

@bot.message_handler(content_types=['text'])
def lalala(message):
    if message.chat.type == 'private':
        if message.text == 'Все варианты':
            for i in range(0,len(mas)):
                p = mas[i].display_place()
                pic=mas[i].get_picture()
                bot.send_message(message.chat.id, f" {p}")
                img = open(f'{pic}', 'rb')
                bot.send_photo(message.chat.id, img)

        elif message.text == 'По площади':

            markup = types.InlineKeyboardMarkup(row_width=2)
            item1 = types.InlineKeyboardButton(">=100", callback_data='good')
            item2 = types.InlineKeyboardButton("<100", callback_data='bad')

            markup.add(item1, item2)

            bot.send_message(message.chat.id, 'какая площадь вас интересует?(в м^2)
', reply_markup=markup)

        elif message.text == 'По цене':

            markup = types.InlineKeyboardMarkup(row_width=4)
            item1 = types.InlineKeyboardButton("<300$", callback_data='small')
            item2 = types.InlineKeyboardButton("300-400$", callback_data='middle')
            item3 = types.InlineKeyboardButton("400-500$", callback_data='middle2')
            item4 = types.InlineKeyboardButton("500-600$", callback_data='large')

            markup.add(item1, item2,item3,item4)

            bot.send_message(message.chat.id, 'какая цена вас интересует?', reply_m
arkup=markup)
        else:
            bot.send_message(message.chat.id, 'Я не знаю что ответить ☹')
```

```
@bot.callback_query_handler(func=lambda call: True)
def callback_inline(call):
    try:
        if call.message:
            if call.data == 'good':
                count=0
                bot.send_message(call.message.chat.id, 'Вот варианты:\n')
                for i in range(0,len(mas)):
                    if(int(mas[i].get_square())>=100):
                        count+=1
                        p = mas[i].display_place()
                        bot.send_message(call.message.chat.id, f" {p}")
                        pic=mas[i].get_picture()
                        img = open(f'{pic}', 'rb')
                        bot.send_photo(call.message.chat.id, img)

                if(count==0):
                    bot.send_message(call.message.chat.id,"К сожалению вариантов не
 нашлось , попробуйте позже")



            elif call.data == 'bad':
                bot.send_message(call.message.chat.id, 'Можем предложить такие вари
анты:')
                count2=0
                for i in range(0,len(mas)):
                    if(int(mas[i].get_square())<100):
                        count2+=1
                        p = mas[i].display_place()
                        bot.send_message(call.message.chat.id, f" {p}")
                        pic=mas[i].get_picture()
                        img = open(f'{pic}', 'rb')
                        bot.send_photo(call.message.chat.id, img)

                if(count2==0):
                    bot.send_message(call.message.chat.id,"К сожалению вариантов не
 нашлось , попробуйте позже")
            elif call.message:
                    if call.data == 'small':
                        count3=0
                        bot.send_message(call.message.chat.id, 'Вот варианты:\n')
                        for i in range(0,len(mas)):
                            if(int(mas[i].get_price())<300):
                                count3+=1
                                p = mas[i].display_place()
                                bot.send_message(call.message.chat.id, f" {p}")
                                pic=mas[i].get_picture()
                                img = open(f'{pic}', 'rb')
                                bot.send_photo(call.message.chat.id, img)

                        if(count3==0):
```

```
                        bot.send_message(call.message.chat.id,"К сожалению вари
антов не нашлось , попробуйте позже")



            elif call.data == 'middle':
                    bot.send_message(call.message.chat.id, 'Можем предложить такие
варианты:')
                    count4=0
                    for i in range(0,len(mas)):
                        if((int(mas[i].get_price())>=300) and (int(mas[i].get_price
()<400))):
                            count4+=1
                            p = mas[i].display_place()
                            bot.send_message(call.message.chat.id, f" {p}")
                            pic=mas[i].get_picture()
                            img = open(f'{pic}', 'rb')
                            bot.send_photo(call.message.chat.id, img)

                    if(count4==0):
                        bot.send_message(call.message.chat.id,"К сожалению варианто
в не нашлось , попробуйте позже")

            elif call.data == 'middle2':
                    bot.send_message(call.message.chat.id, 'Можем предложить такие
варианты:')
                    count5=0
                    for i in range(0,len(mas)):
                        if(int(mas[i].get_price())>=400 and int(mas[i].get_price()<
500)):
                                count5+=1
                                p = mas[i].display_place()
                                bot.send_message(call.message.chat.id, f" {p}")
                                pic=mas[i].get_picture()
                                img = open(f'{pic}', 'rb')
                                bot.send_photo(call.message.chat.id, img)

                    if(count5==0):
                        bot.send_message(call.message.chat.id,"К сожалению варианто
в не нашлось , попробуйте позже")

            elif call.data == 'large':
                    bot.send_message(call.message.chat.id, 'Можем предложить такие
варианты:')
                    count6=0
                    for i in range(0,len(mas)):
                            if(int(mas[i].get_price())>=500 and int(mas[i].get_pric
e()<600)):
                                count6+=1
                                p = mas[i].display_place()
                                bot.send_message(call.message.chat.id, f" {p}")
                                pic=mas[i].get_picture()
                                img = open(f'{pic}', 'rb')
                                bot.send_photo(call.message.chat.id, img)
```

```
                    if(count6==0):
                        bot.send_message(call.message.chat.id,"К сожалению варианто
в не нашлось , попробуйте позже")
    except Exception as e:
        print(repr(e))


        # show alert
        bot.answer_callback_query(callback_query_id=call.id, show_alert=False,
        text="Спасибо что воспользовались нашим ботом)))")
# RUN
bot.polling(none_stop=True)
```