**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**ОТЧЕТ о выполнении**

**ЛАБОРАТОРНОЙ РАБОТЫ №1**

**по дисциплине “Основы облачных технологий”**

Работу выполнили студентки 4 курса 4ИТ гр. \_\_\_\_\_\_\_\_\_\_\_\_\_ Махалин С.А.

Работу проверил канд. техн. наук, доц.\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Полупанов А. А.

Краснодар 2023

Тема: Разработка Telegram-бота на платформе Yandex Cloud с тестированием на знаменитость по базовым внешним признакам.

Цель: Изучение теоретических основ облачных вычислений, внутренней структуры и практической реализации, и прикладных примеров использования облачных вычислений и веб-сервисов Yandex Cloud, а именно Cloud Functions, Managed Service for PostgreSQL, Object Storage и API Gateway.

Задание: Разработка Telegram-бота с использованием облачных сервисов.

Постановка задачи: разработать чат-бота с функционалом тестирования схожести на знаменитость.

# Аккаунт Yandex Cloud

Платформа Yandex Cloud предоставляет несколько вариантов интерфейсов для пользователей, таких как консоль управления и командная строка. Для доступа к любому из этих интерфейсов необходимо иметь собственный пользовательский аккаунт.

Платежный аккаунт используется для идентификации пользователя, который оплачивает услуги и ресурсы, предоставляемые Yandex Cloud. Даже если ваши цели ограничиваются использованием бесплатных сервисов, все равно необходимо создать платежный аккаунт, так как на него могут быть начислены гранты и промокоды.

Чтобы активировать платежный аккаунт, необходимо привязать банковскую карту. Это необходимо для подтверждения вашей личности и исключения возможности использования аккаунта роботами. При привязке карты будет временно удержана небольшая сумма средств, которая затем будет возвращена. Этот шаг предназначен для проверки подлинности карты.

Средства будут списываться с вашей карты только в случае вашего явного разрешения на платное использование сервисов и ресурсов Yandex Cloud.

# Создание платежного аккаунта

Даже если вы планируете использовать только бесплатные сервисы, необходимо создать платежный аккаунт. После создания аккаунта будем начислен бонусный грант размером в 3000 бонусов. Для начала процесса необходимо выполнить вход с использованием Яндекс ID и зайти в консоль управления. Затем, чтобы создать свой первый платежный аккаунт, выполните следующие шаги:

1. Перейдите в раздел "Биллинг" в консоли управления;
2. На странице "Аккаунты" нажмите кнопку "Создать аккаунт" и заполните необходимые данные;
3. Если у вас есть список доступных плательщиков в разделе "Плательщики", вы можете выбрать одного из них или добавить нового;
4. Для добавления нового плательщика выберите тип плательщика и привяжите карту.

Для проверки валидности карты будет временно заблокирована небольшая сумма на вашем счете. Эти средства не будут списаны и станут доступными снова после завершения проверки и разблокировки. Время разблокировки несписанных средств зависит от правил вашего банка.

Если это ваш первый платежный аккаунт в Yandex Cloud, у вас будет возможность подключить пробный период. После завершения пробного периода использование ресурсов будет приостановлено, и для возобновления работы потребуется перейти на платную версию. Если вы не подключаете пробный период на данном этапе, аккаунт будет создан с платным использованием, после использования стартового гранта. После выполнения всех необходимых шагов, нажмите кнопку "Создать".

Обратите внимание, что все плательщики в рамках одного пользовательского аккаунта должны быть либо резидентами РФ, либо резидентами РК, либо резидентами других стран.

После всех проделанных операций появится информация о платежном аккаунте (рис 1)

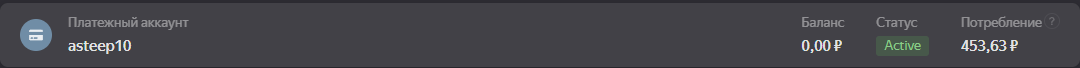


Рисунок 1 – платежный аккаунт

# Работа с Yandex Resource Manager

Этот сервис предназначен для управления ресурсами в Yandex Cloud, такими как фукнкции, каталоги и другие облачные ресурсы. Когда у вас есть доступ к Yandex Cloud, вы получаете собственное рабочее пространство, которое называется облаком. Внутри облака вы можете создавать каталоги, в которых будут размещаться различные ресурсы, такие как виртуальные машины, диски и другие. При создании ресурса вы указываете, в каком каталоге он будет расположен. Важно отметить, что вложенные каталоги в данный момент не поддерживаются.

Resource Manager предоставляет стандартную модель управления ресурсами, которая используется в большинстве сервисов Yandex Cloud. По умолчанию все ресурсы в облаке изолированы от пользователей, которые не имеют доступ к облаку. Владелец облака имеет полный контроль над правами доступа как к облаку в целом, так и к ресурсам внутри него.

Права доступа к ресурсам в облаке наследуются: права, предоставленные на облако, автоматически распространяются на все ресурсы внутри облака, а права на каталог распространяются на все ресурсы в этом каталоге. Однако некоторые типы ресурсов не создаются в каталогах и имеют собственную логику проверки прав доступа. Например, при управлении доступом к сервисным аккаунтам проверяются права доступа к самому сервисному аккаунту.

Облако представляет собой изолированное пространство, внутри которого создаются каталоги. При создании ресурса внутри облака, доступ к этому ресурсу по умолчанию закрыт для всех, кроме участников и владельцев данного облака.

# Создание нового облака

1. При регистрации автоматически создается облако с названием "cloud-<ваш Яндекс ID>". После привязки платежного аккаунта, появляется возможность создать дополнительное облако. Для убеждения в том, что платежный аккаунт активен, или находится в статусе TRIAL\_ACTIVE, можно посетить страницу биллинга. Если у вас еще нет платежного аккаунта, также можно запросить администратора назначить вам роль billing.accounts.member на соответствующий платежный аккаунт.

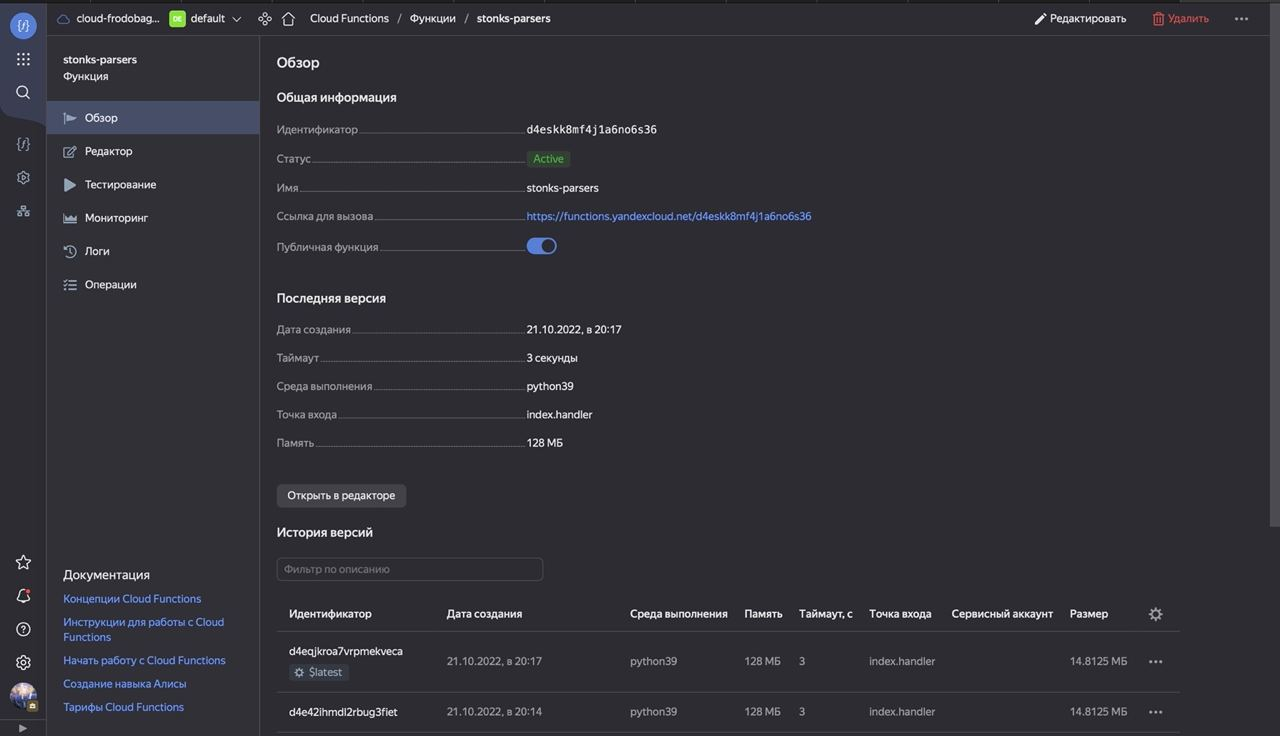


Рисунок 2— статус Active

1. На стартовой странице консоли управления необходимо нажать значок напротив своего биллинг-аккаунта.
2. Выбрать *Создать облако*.
3. Ввести имя облака.
4. Указать организацию, к которой будет привязано облако. Если еще нет организации, она будет создана автоматически.
5. Щелкните кнопку "Создать". Вы будете перенаправлены на страницу каталога "default" в новом облаке.

При создании облака автоматически устанавливается роль владельца для этого облака. Владелец имеет возможность создавать новые каталоги и ресурсы в этом облаке, а также управлять правами доступа к ним.

Каталог в данном контексте - это пространство, в котором можно создавать и организовывать ресурсы Yandex Cloud. Подобно каталогам в файловой системе, каталоги в Yandex Cloud упрощают управление ресурсами. Вы можете группировать ресурсы в каталоги по различным критериям, таким как тип ресурса, проект, отдел или другие параметры.

Вы также можете управлять правами доступа ко всем ресурсам в каталоге. Например, если у вас есть сотрудники, работающие исключительно с виртуальными машинами, вы можете создать каталог, содержащий только виртуальные машины, и предоставить этим сотрудникам доступ к этому каталогу.

Вы можете просматривать содержимое каталога в консоли управления. Если вам нужно узнать, к какому каталогу относится определенный ресурс, вы можете воспользоваться API или CLI, выполнив метод "Get" для этого ресурса (команду "get" в CLI).

**Создание каталога**

1. В консоли управления, выберите облако и нажмите "Создать каталог."
2. Введите название каталога.
3. По желанию, добавьте описание каталога.
4. Выберите опцию "Создать сеть по умолчанию." Это приведет к созданию сети с подсетями в каждой зоне доступности и созданию группы безопасности по умолчанию, которая разрешает весь сетевой трафик.
5. Нажмите "Создать."

# Создание облачной базы данных Yandex Cloud

Для создания базы данных PostgreSQL в Yandex Cloud, выполните следующие шаги:

1. Перейдите в раздел "Сервисы": После входа в аккаунт, перейдите в раздел "Сервисы" в верхнем меню.
2. Выберите "Базы данных": В разделе "Сервисы" выберите "Базы данных."
3. Нажмите кнопку "Создать кластер PostgreSQL".
4. Заполните необходимую информацию о кластере, такую как название, регион, версия PostgreSQL и ресурсы (CPU, RAM, и др.), пример (рис 2)



Рисунок 3 – характеристики кластера БД

1. Укажите пароль для администратора базы данных.
2. Нажмите "Создать."
3. Необходимо дождаться завершения создания: Создание кластера может занять некоторое время, в зависимости от выбранных параметров.
4. После создания кластера, перейдите в его настройки.
5. В разделе "Доступы" можно добавить пользователей и настроить их права доступа.
6. В разделе "Базы данных" можно создать новые базы данных или управлять существующими.

В разделе SQL вы можете взаимодействовать с базой данных, если вы указали возможность управления из консоли.

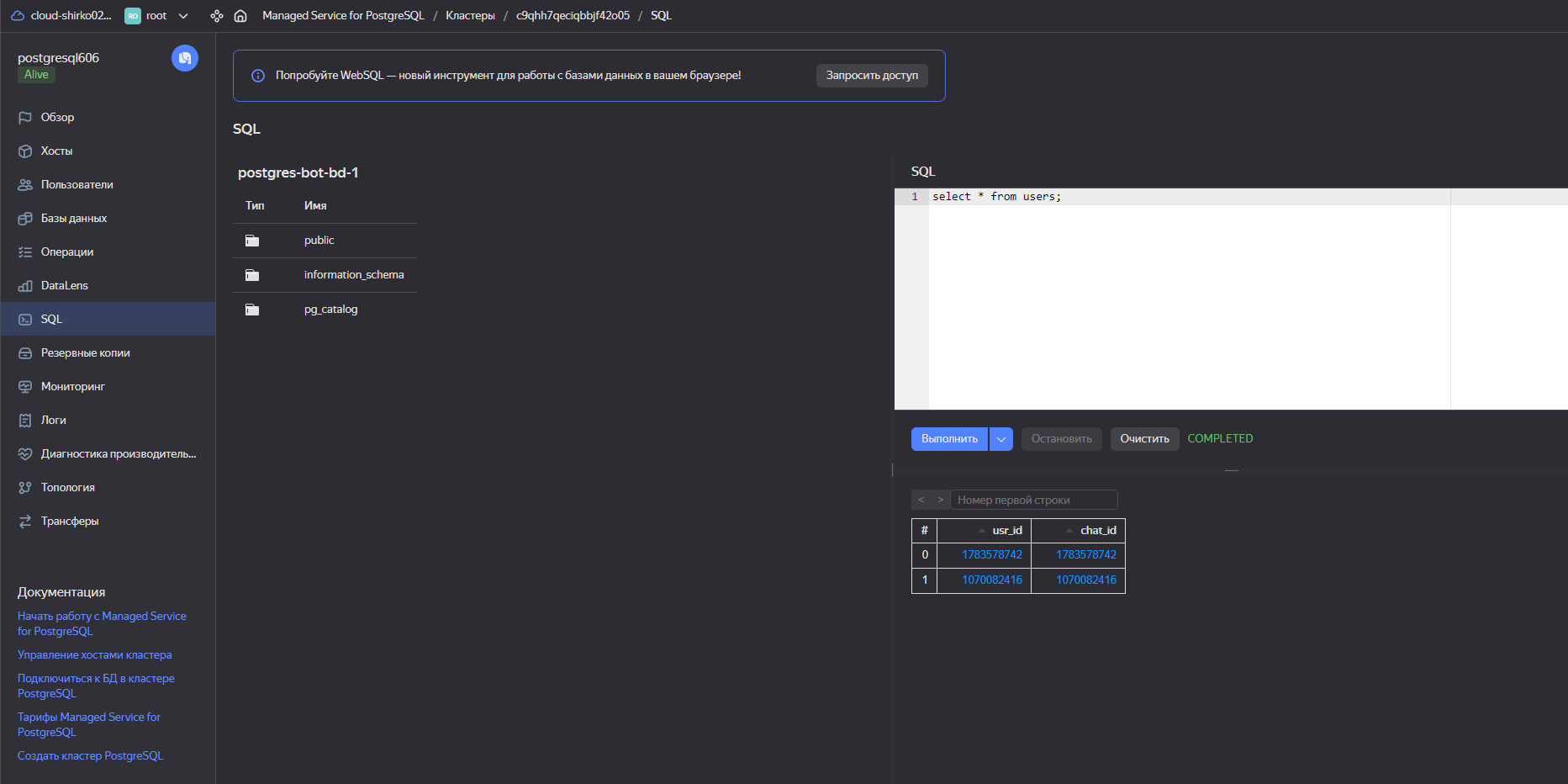


Рисунок 4 – управление БД из консоли

Для подключения к БД можно использовать любой клиент PostgreSQL (например, pgAdmin, psql) или библиотеки для программирования для подключения к вашей базе данных, используя информацию о подключении через SSL, предоставленную в настройках кластера.

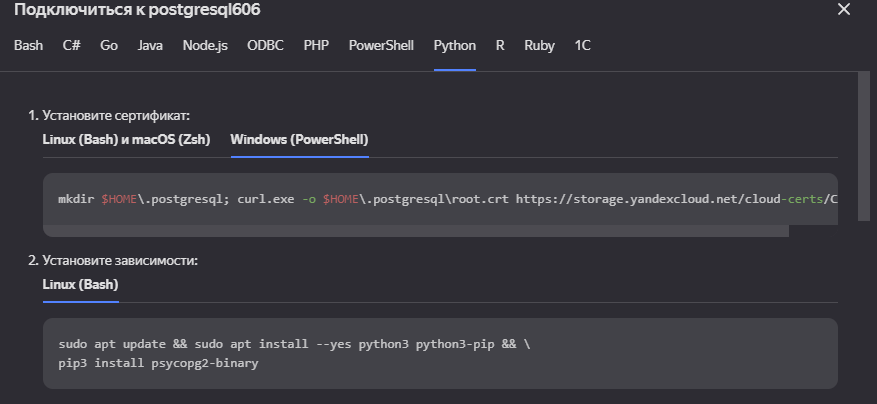


Рисунок 5 – способ подключения к БД на Python

Для создания Cloud Functions нужно:

1. Перейти на главную страницу, выбрать в списке «Все сервисы» «Cloud Functions»
2. Далее нажать «Создать функцию»
3. Написать имя функции, нажать кнопку «Создать»
4. В созданной функции переключить метку на «Публичная функция»
5. Перейти во вкладку «Редактор»
6. Добавить код, либо добавить zip-архив

Для создания API Gateway, на который бот будет присылать запрос, редиректящийся на Cloud Functions, необходимо:

# Создание облачной базы данных в VK Cloud

Для создания базы данных PostgreSQL в VK Cloud прежде всего необходимо зарегистрироваться на данном портале. Процесс регистрации полностью аналогичен Yandex Cloud, за исключением звонка оператора с поздравлением к присоединению к порталу.

VK Cloud предоставляет множество облачных решений, начиная от облачных вычислений и заканчивая работой с 1С.

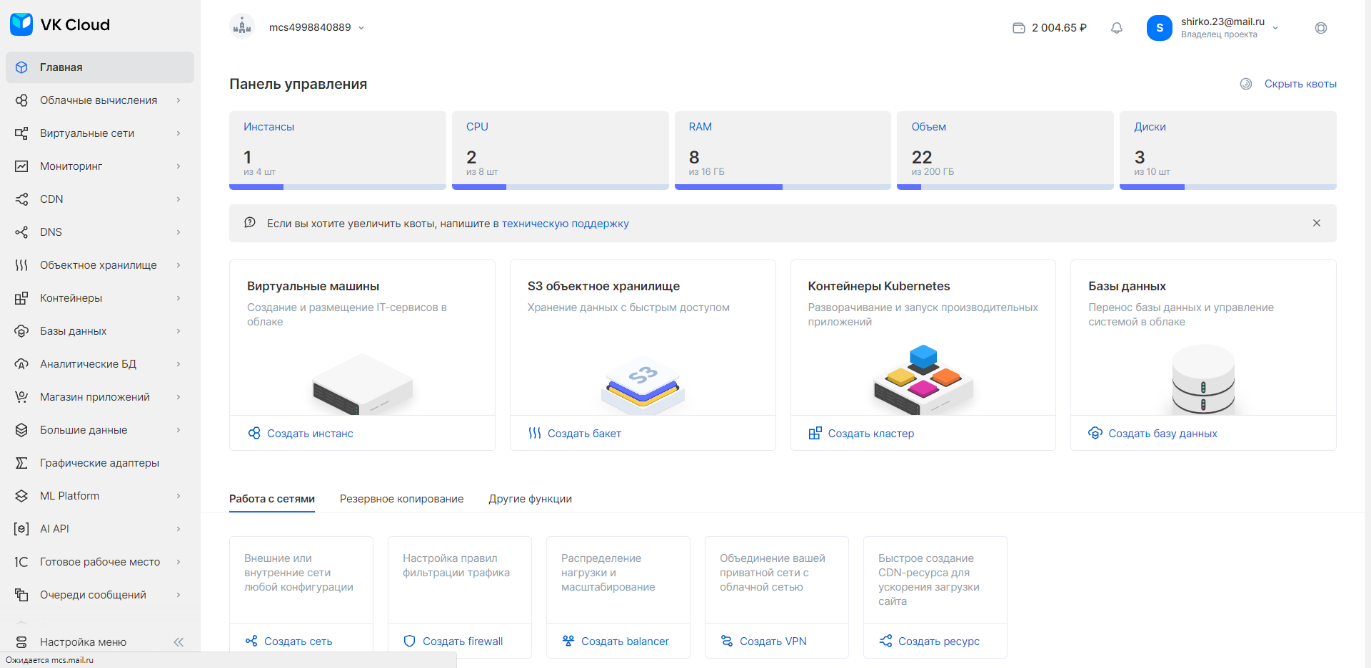


Рисунок 3 – возможности VK Cloud

Тем не менее в данной работе интерес представляет создание отдельного инстанса, а именно облачной базы данных.

Для того чтобы создать БД в VK Cloud необходимо выполнить следующие шаги:

1. На главной странице необходимо перейти в раздел «Базы Данных»
2. Выбираем необходимый тип БД, в нашем случае PostgreSQL и тип конфигурации
3. Настраиваем БД согласно требованиям

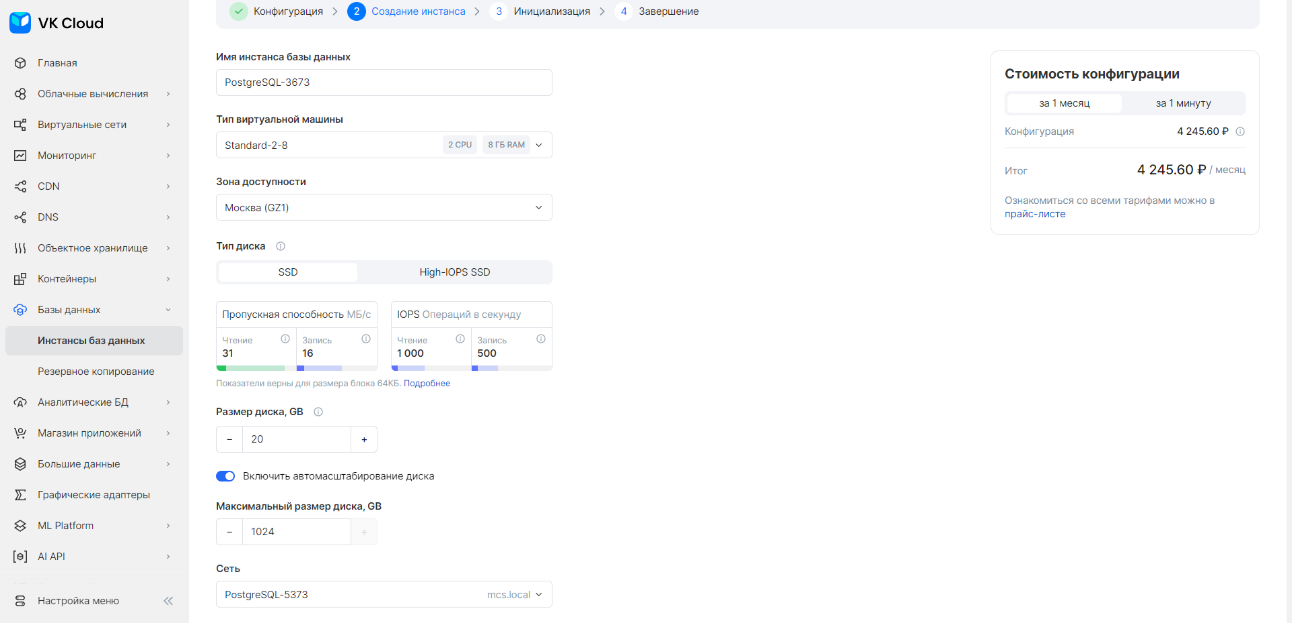


Рисунок 4 – настройка подключаемой БД

1. После создания нового инстанса – краткая информация об БД появится в соответствующем разделе. Где необходимо будет дополнительно настроить внешний-ip

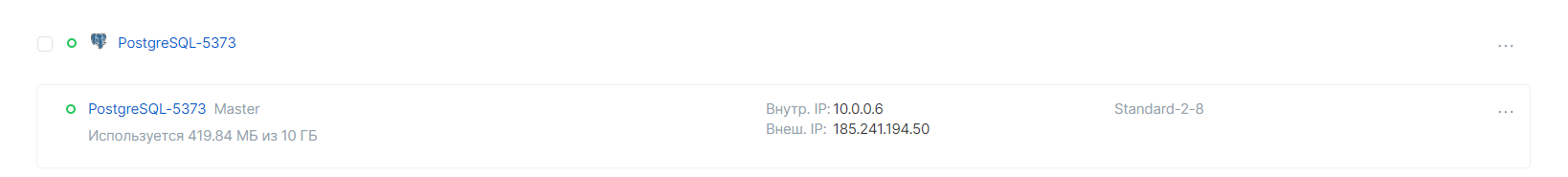


Рисунок 5 – краткая информация о БД

1. При открытии инстанса можно получать более полную информацию, включая параметры СУБД и способы подключения к ней.

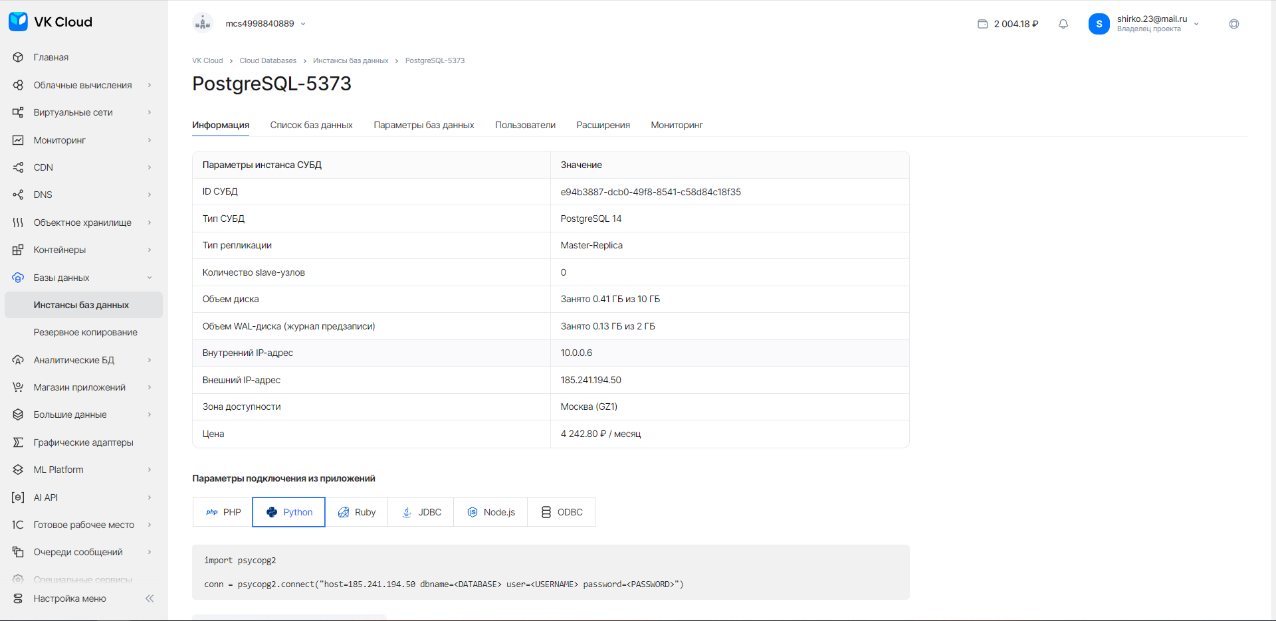


Рисунок 6 – подробная информация о БД

Для взаимодействия с БД можно использовать любой GUI PostgreSQL (например, pgAdmin, psql) или среду разработки.

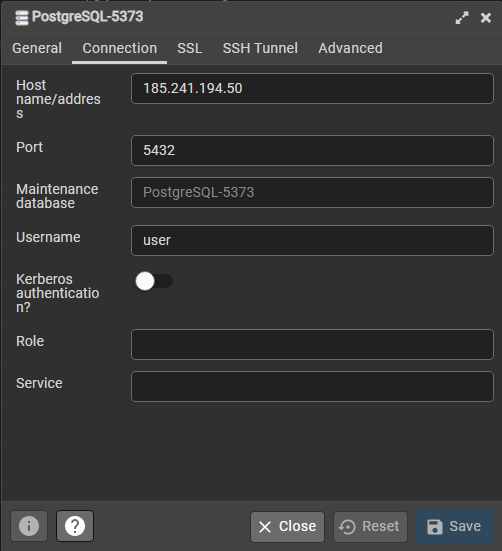


Рисунок 7 – пример подключения к БД в pgAdmin

# Разработка чат-бота

# Базовое создание облачной функции

Когда у вас уже есть отдельное рабочее пространство, или, иными словами, облако, следующим шагом будет создание каталога внутри этого облака, где вы сможете организовать ресурсы.

Для начала настройки необходимо открыть консоль управления и перейти к сервису *Cloud Functions*. После этого выбрать нужную функцию и перейдите на вкладку "Редактор". Если функция пока что пуста, вы должны выбрать среду выполнения и нажать кнопку "Продолжить". В разделе "Код функции" выполните следующие шаги:

1. Выберите среду выполнения, например, Python 3.9.
2. Выберите метод редактирования функции, который предпочтительно использовать, например, "Редактор кода".
3. Перейдите в панель навигации и выберите существующий файл или создайте новый файл или архив файлов, если это необходимо.

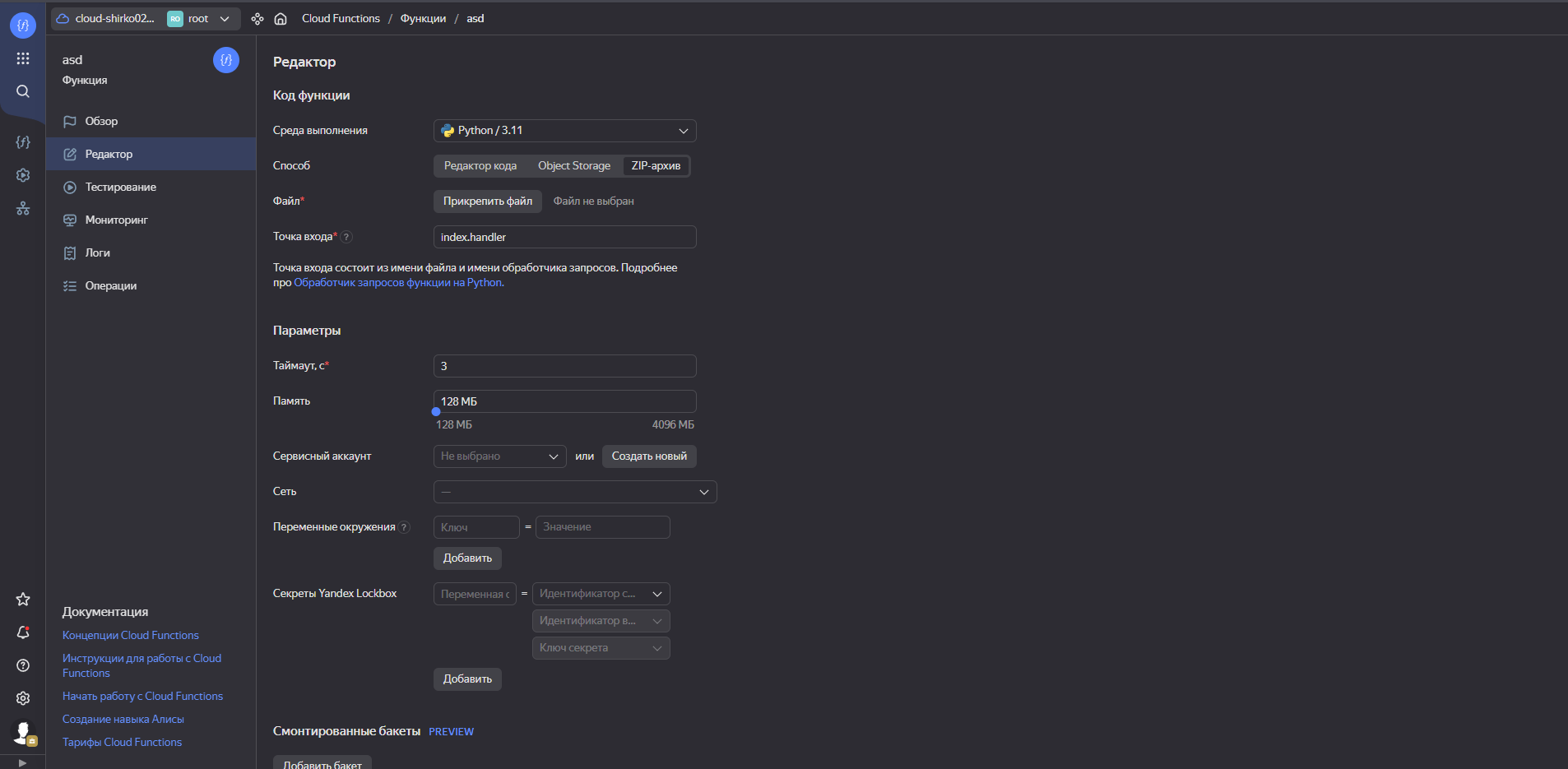


Рисунок 6 - инициализация облачной функции

1. Укажите точку входа, которая представляет собой имя функции, которая будет вызываться как обработчик. Формат: "<имя файла без расширения>.<имя обработчика>". Например, index.Handler.

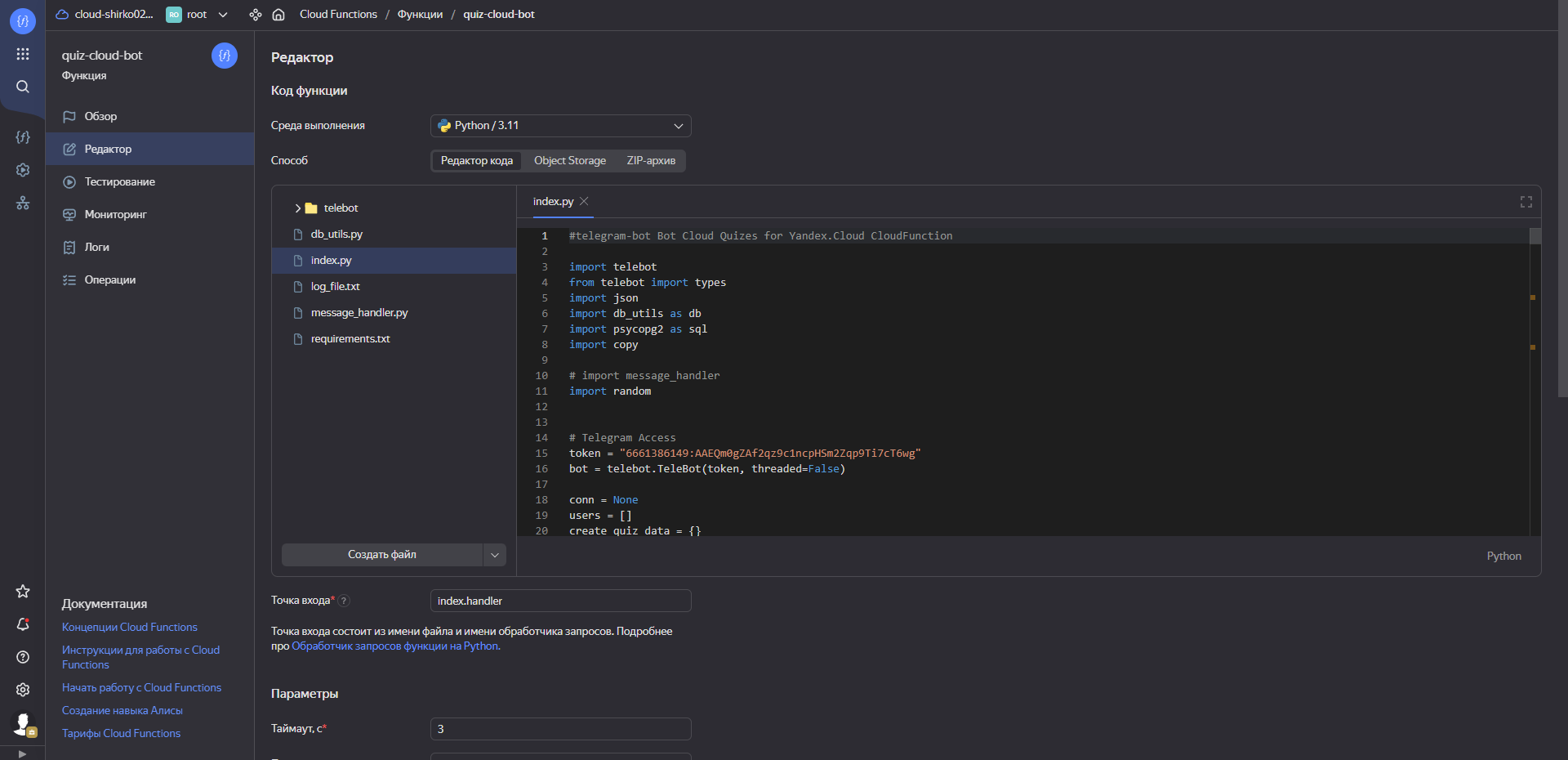


Рисунок 7 – редактор Cloud Function

1. Если программа должна подключать дополнительные зависимости, то создайте дополнительный файл requirements.txt с описанием необходимой библиотеки и её версии.

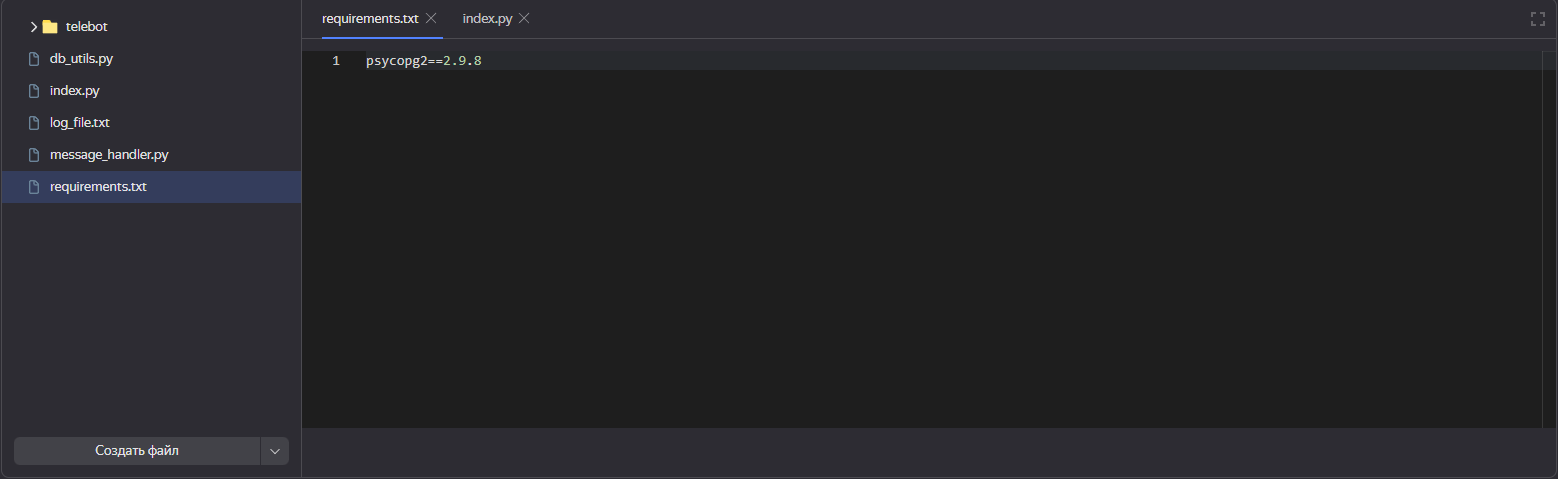


Рисунок 8 – файл описания библиотек

1. Если приложение должно взаимодействовать с БД, предложенными решениями Yandex Cloud, необходимо перейти в раздел «подключения к БД» и нажать кнопку «Создать подключение». Также в редакторе функции необходимо указать сервисный аккаунт с ролью serverless.mdbProxies.user – необходимой для взаимодействия с БД без использования дополнительных сертификатов в serverless приложении.

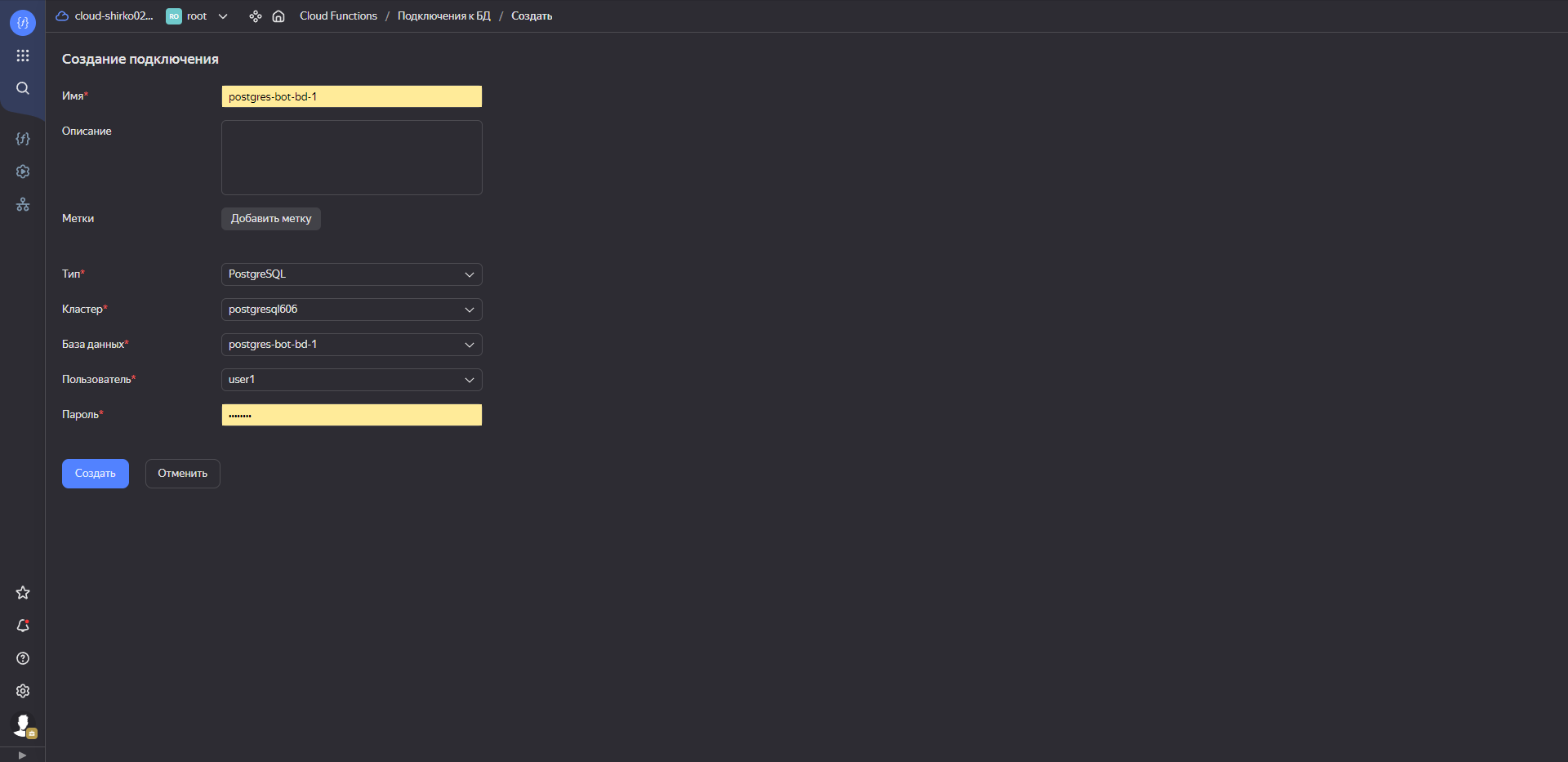


Рисунок 9 – подключение БД к Cloud Function

1. В блоке ***Параметры* в редакторе функции**:
   * Указываем таймаут.
   * Указываем объем памяти.
   * (опционально) Выбираем или создаём сервисный аккаунт. При добавлении сервисного аккаунта для функции можно получать его IAM-токен из контекста функции.
   * (опционально) Добавляем переменные окружения.
2. После внесения необходимых изменений необходимо нажать на кнопку «***Создать версию»***.

Формируется ссылка для вызова.

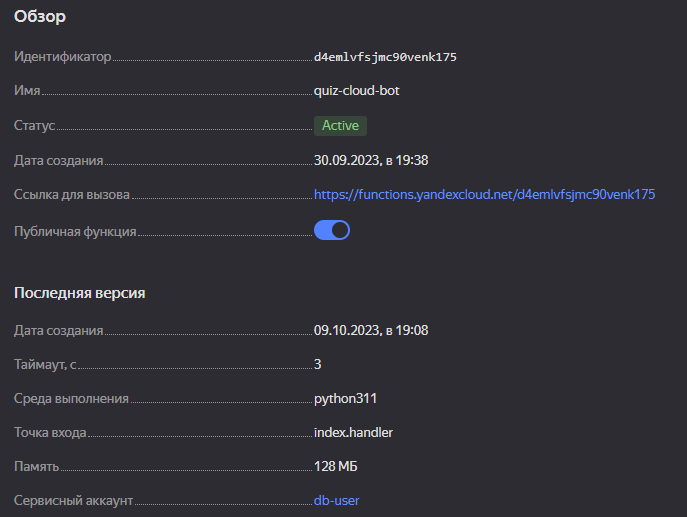


Рисунок 10 – создание рабочей версии

В дальнейшем используем **э**ту ссылку для управления чат-ботом.

# Создание чат-бота

BotFather. Настройка бота. Сначала необходимо зарегистрировать саму оболочку бота в самом мессенджере, получить присвоенный ей ID, который является также токеном. В Телеграмм помощь при такой регистрации оказывает виртуальный помощник @BotFather.

1. ввод команды /start для запроса инструкции;
2. отправка установки /newbot;
3. @BotFather предлагает придумать имя для создаваемого бота. Оно должно заканчиваться словом «bot»;
4. далее помощник присылает токен нового виртуального пользователя и url для добавления его в список контактов.

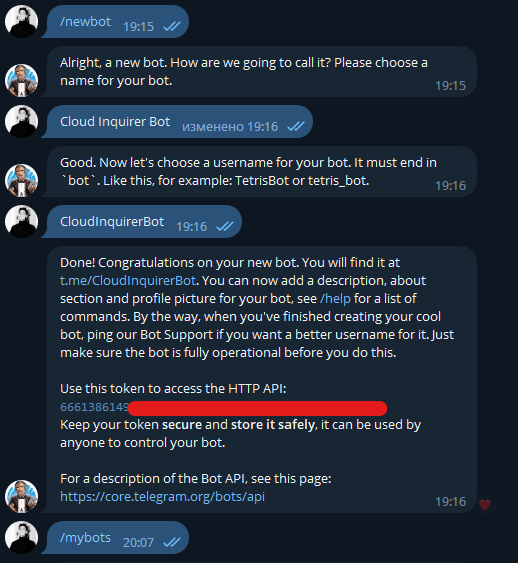


Рисунок 11 – регистрация чат-бота

Так же необходимо установить веб-хук для того, что бы бот сам реагировал на изменения в чатах с пользователями. Для этого был написан специальный скрип на языке Python (рис 12).

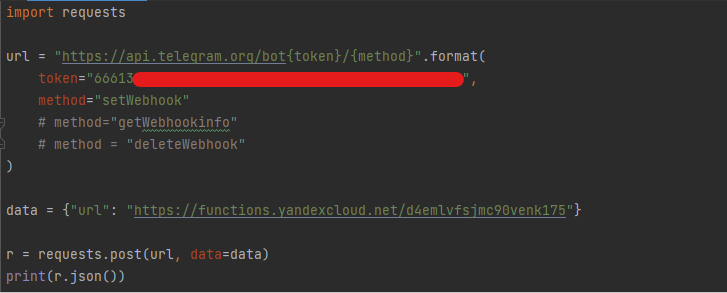


Рисунок 12 – скрипт для установки веб-хука на чат-бота

Заметим, что в словаре data указывается непосредственно ссылка на облачную функцию – обработчик чат-бота.

После выполнения скрипта, получаем ответ (рис 13).

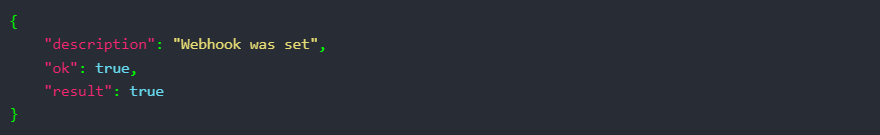


Рисунок 13 – успешная установка веб-хука

# Описание работы чат-бота

Для работы чат-бота в Telegram была использована библиотека telebot – официальное API для работы с ботами.

Для работы с БД были написаны функции, представляющие собой специальные запросы в БД с целью получения определенных данных.

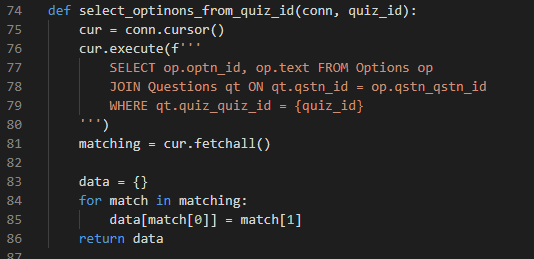


Рисунок 14 – пример функции-обращения к БД

Сама структура программы бота состоит из следующих частей:

1. Функция – Handler - является точкой входа в облачную функцию, в которой происходит базовая инициализация данных и открытия канала связи с БД;
2. Start\_helper – функция старта работы с чат-ботом, в которой происходит регистрация нового пользователя в программе;
3. Answer\_message – функция построения диалога с пользователем, использует специальные состояния для определения следующего логического ответа в диалоге с пользователем;
4. Others – функции необходимы для правильной работы логики чат-бота-опросника.

# Функционал чат-бота

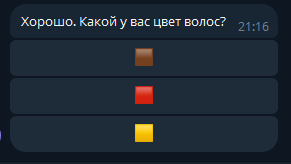
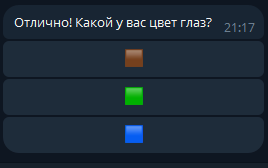
# Стартовая команда для начала работы бота-опросника:

# 

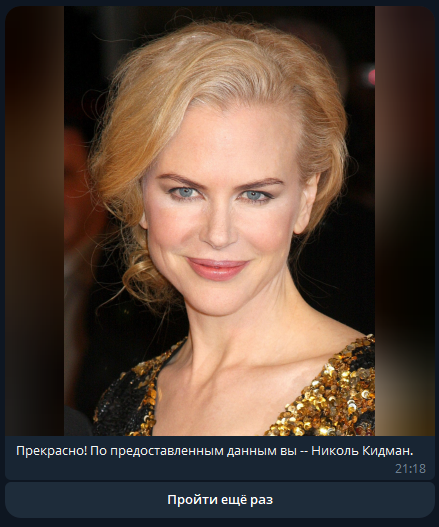
# Рисунок 15 – приветствие

# Далее необходимо пройти опрос:

# 

Рисунок 16 – пол  
  
  
Рисунок 17 – цвет волос  
  
  
Рисунок 18 – цвет глаз

В конце мы получим вывод о том, на какую знаменитость похожи:

  
Рисунок 19 – результат

Результаты прохождения хранятся в БД, поэтому в следующий раз, когда будет написано /start, выведется старый результат:

  
Рисунок 20 – старый результат тестирования

# Вывод

Рассмотрели облачную платформу Yandex Cloud и VK Cloud. Изучили теоретическую основу и выявили особенности внутренней структуры. Разработали чат-бота опросника с использованием облачных функций, базы данных и других сервисов.

# Приложение 1

Ссылка на исходники приложения: https://github.com/xuoda42/cloud\_program/tree/master

index.py

def handler(event, \_):

    message = telebot.types.Update.de\_json(event['body'])

    bot.process\_new\_updates([message])

    return {

        'statusCode': 200,

        'body': '!',

    }