

**КОЛЛЕДЖ КОСМИЧЕСКОГО МАШИНОСТРОЕНИЯ И ТЕХНОЛОГИЙ**

**Курсовой проект**

**По МДК 01.02 Прикладное программирование**

**РАЗРАБОТКА ДИСКОРД БОТА «РЮ»**

Пояснительная записка

КП.09.02.03.22.07ПЗ

Обучающийся группы П1-19 Дроздов Н.А.

Руководитель курсового проекта Кочетков С. С.

Результат защиты\_\_\_\_\_\_\_\_\_\_\_

**Королёв, 2022**



**КОЛЛЕДЖ КОСМИЧЕСКОГО МАШИНОСТРОЕНИЯ И ТЕХНОЛОГИЙ**

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | |  | «УТВЕРЖДАЮ»  Заместитель директора по учебной работе  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.В. Антропова  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г. | |

**ЗАДАНИЕ**

**на курсовой проект**

МДК 01.02 Прикладное программирование

обучающийся специальности 09.02.03 Программирование в компьютерных системах

группы П1-19, 3 курса **Дроздов Никита Андреевич**

Тема курсового проекта **«Разработка дискорд бота «Рю»»**

Утверждена приказом от «26» апреля 2022 г. № 01-05/916

Содержание проекта:

Введение.

Глава 1. Анализ и классификация различных типов Discord ботов.

Глава 2. Разработка и проектирование Discord бота «Рю» на Python.

Глава 3. Анализ работы Discord бота «Рю».

Заключение.

Список использованной литературы и интернет - ресурсов.

Приложения.

Презентация.

Срок выполнения курсового проекта 23 июня 2022 года

Председатель цикловой комиссии \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. А. Эшанов

Руководитель курсового проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С. С. Кочетков подпись

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.А. Дроздов

подпись студента

Дата выдачи задания «26» апреля 2022 года

**Введение.**

Целью моего проекта является создание бота-помощника для мессенджера – Discord. Данное ПО в этом курсовом проекте призвано упростить жизнь обычным людям и геймерам (для людей, которые проводят много времени в играх/занимаются играми профессионально), предоставляя дополнительный функционал, который по тем или иным причинам недоступен в основной программе.

Актуальность темы курсовой работы обусловлено все сильнее развивающемся интернетом, программ основанных на передаче данных через сеть интернет и иными способами передачи данных. Так же, не на всех серверах присутствует качественный бот, который имеет большой функционал, на многих его вообще нет. Исходя из этого видно, что данное приложение будет очень актуально как геймерам, так и простым людям, которым необходимо что-либо обсудить через интернет.

# Глава 1. Анализ и классификация различных типов Discord ботов.

## 1.1 Исследование предметной области.

Область данного программного решения (боты для серверов Discord) довольно популярна среди программистов. Популярна она тем, что с помощью API-интерфейса можно взаимодействовать с приложением, и предоставлять пользователю дополнительный функционал, который в самом клиенте не представлен.

Сам Discord – это бесплатный мессенджер, изначально ориентированный на пользователей компьютерных игр. Эта программа появилась не так давно, но благодаря своей многофункциональности она сумела за короткий промежуток времени набрать огромное количество поклонников. Ежемесячно к системе Discord подключаются около 2 млн. пользователей со всех уголков мира.

У него есть аналоги, но на данный момент они потеряли свою былую популярность (Skype, TeamSpeak).

Целью данного курсового проекта является разработка Discord-бота который будет выполнять следующие задачи:

* Взаимодействие с пользователями через чат сервера;
* Выполнение администраторских функций;

Сам программный продукт нацелен на аудиторию игроков и администраторов, которые участвуют в управлении своего Discord-сервера.

Готовность пользователей к данному программному решению можно оценить как высокую, так как, как было указано выше – данные боты имеют высокую популярность среди пользователей, и для них не составит труда выучить простые команды для взаимодействия с ботом.

## 1.2 Описание средств разработки.

Данная программа была написана на высокоуровневом языке программирования Python с помощью JetBrains PyCharm. PyCharm — [интегрированная среда разработки](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) для языка программирования [Python](https://ru.wikipedia.org/wiki/Python). Предоставляет средства для анализа кода, графический отладчик, инструмент для запуска юнит-тестов, разрабатываемая компанией [JetBrains](https://ru.wikipedia.org/wiki/JetBrains).

Программирование выполнялось на языке Python с использованием Discord API.

Python - [высокоуровневый язык программирования](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) общего назначения с [динамической](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) [строгой](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D0%BE%D0%B3%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости [кода](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) и его качества, а также на обеспечение переносимости написанных на нём программ. Разработан нидерландским программистом Гвидо ван Россумом в 1989 году. Язык является полностью [объектно-ориентированным](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) в том плане, что всё является [объектами](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)). Необычной особенностью языка является выделение [блоков кода](https://ru.wikipedia.org/wiki/%D0%91%D0%BB%D0%BE%D0%BA_%D0%BA%D0%BE%D0%B4%D0%B0) пробельными отступами. [Синтаксис](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BD%D1%82%D0%B0%D0%BA%D1%81%D0%B8%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Сам же язык известен как [интерпретируемый](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BF%D1%80%D0%B5%D1%82%D0%B8%D1%80%D1%83%D0%B5%D0%BC%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) и используется в том числе для написания [скриптов](https://ru.wikipedia.org/wiki/%D0%A1%D0%BA%D1%80%D0%B8%D0%BF%D1%82).

Discord API – это программный интерфейс приложения, обращение к которому напрямую значительно упрощает процесс разработки ботов.

# Глава 2. Разработка и проектирование Discord бота «Рю» на Python.

## 2.1 Постановка задачи.

Для взаимодействия c самим ботом используются команды, вводимые в любой текстовый канал сервера. Все команды начинаются с префикса «!». Это необходимо для того, чтобы бот мог отличать команды от обычных сообщений пользователей сервера. С помощью команд должна быть реализованы функции, которые работают со списком пользователей на сервере, административные команды.

## 2.2 Анализ требований ПО.

Был проведен анализ требований программного обеспечения, и составлена диаграмма вариантов использования, она изображена на рисунке 1.

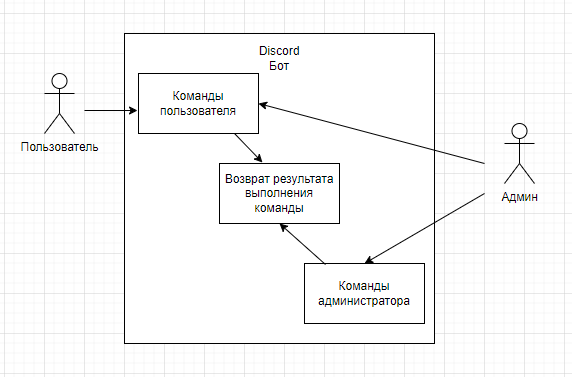


Рисунок 1 - Диаграмма вариантов использования

## 2.3 Планирование ПО.

Было проведено планирование программного обеспечения, и была составлена структурная схема программы, изображенная на рисунке 2.

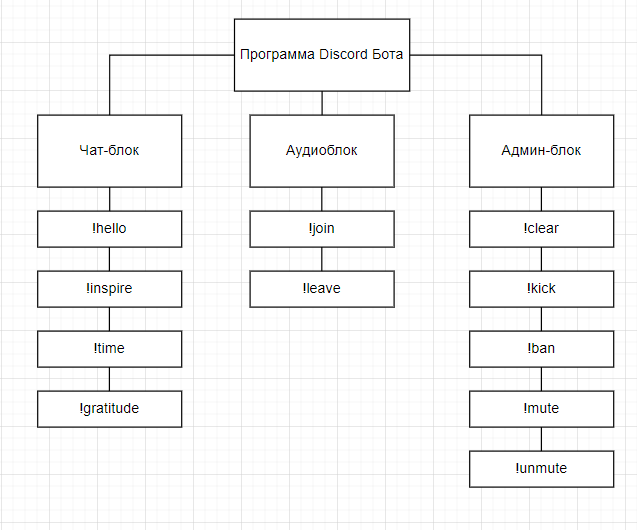


Рисунок 2 - Структурная схема программы

Так же была составлена функциональная схема, наглядно показывающая принцип работы программы, представленная на рисунке 3.

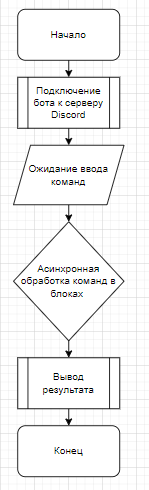


Рисунок 3 - Функциональная схема программы

# Глава 3. Анализ работы Discord бота «Рю».

В начале нужно создать объект бота для Discord, для этого нужно перейти на сайт для разработки ботов Discord – Discord Developer Portal, авторизоваться и нажать на кнопку «New Application», как показано на рисунке 4.

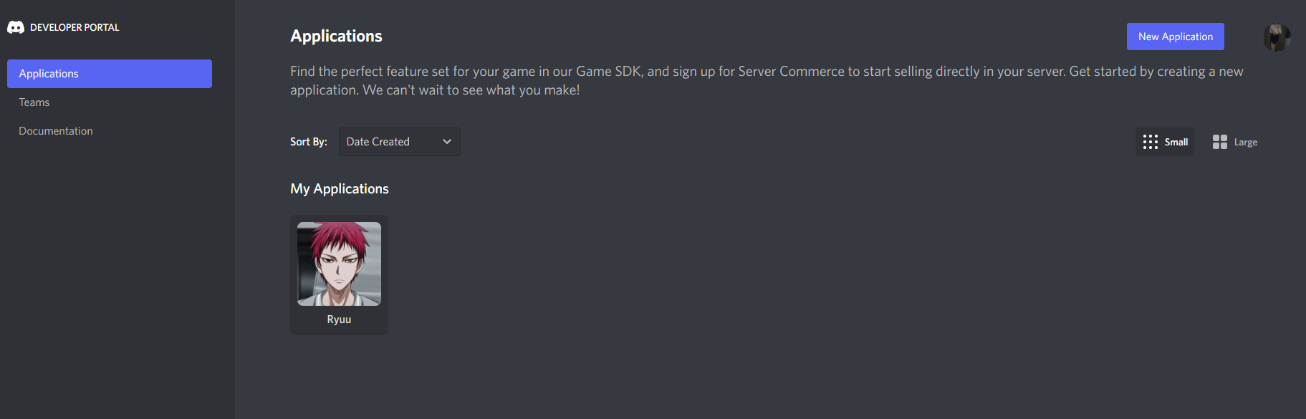


Рисунок 4 - Приложения

Ввести имя бота, как показано на рисунке 5.

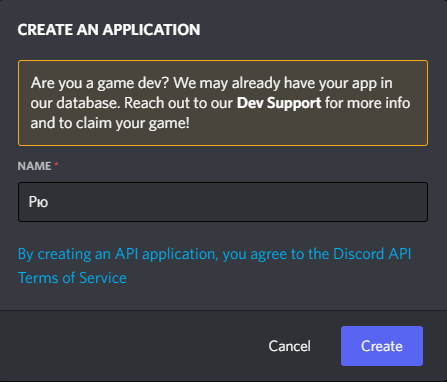


Рисунок 5 - Ввод имени бота

После этого нужно нажать «Create» и перейти на страницу бота, изображенного на рисунке 6.

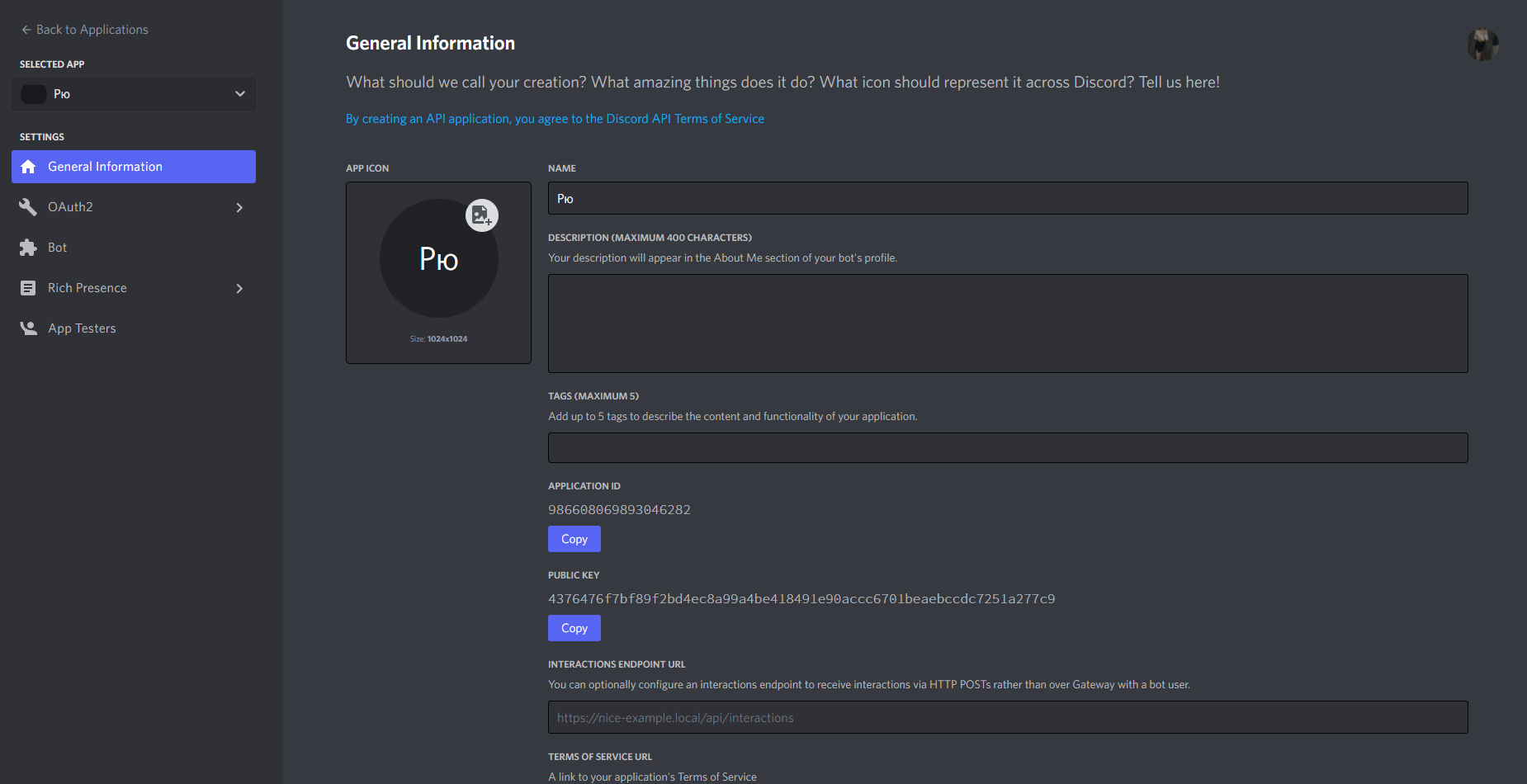


Рисунок 6 - Страница бота

Далее необходимо перейти во вкладку «Bot» и нажать кнопку «Add Bot», как показано на рисунке 7.

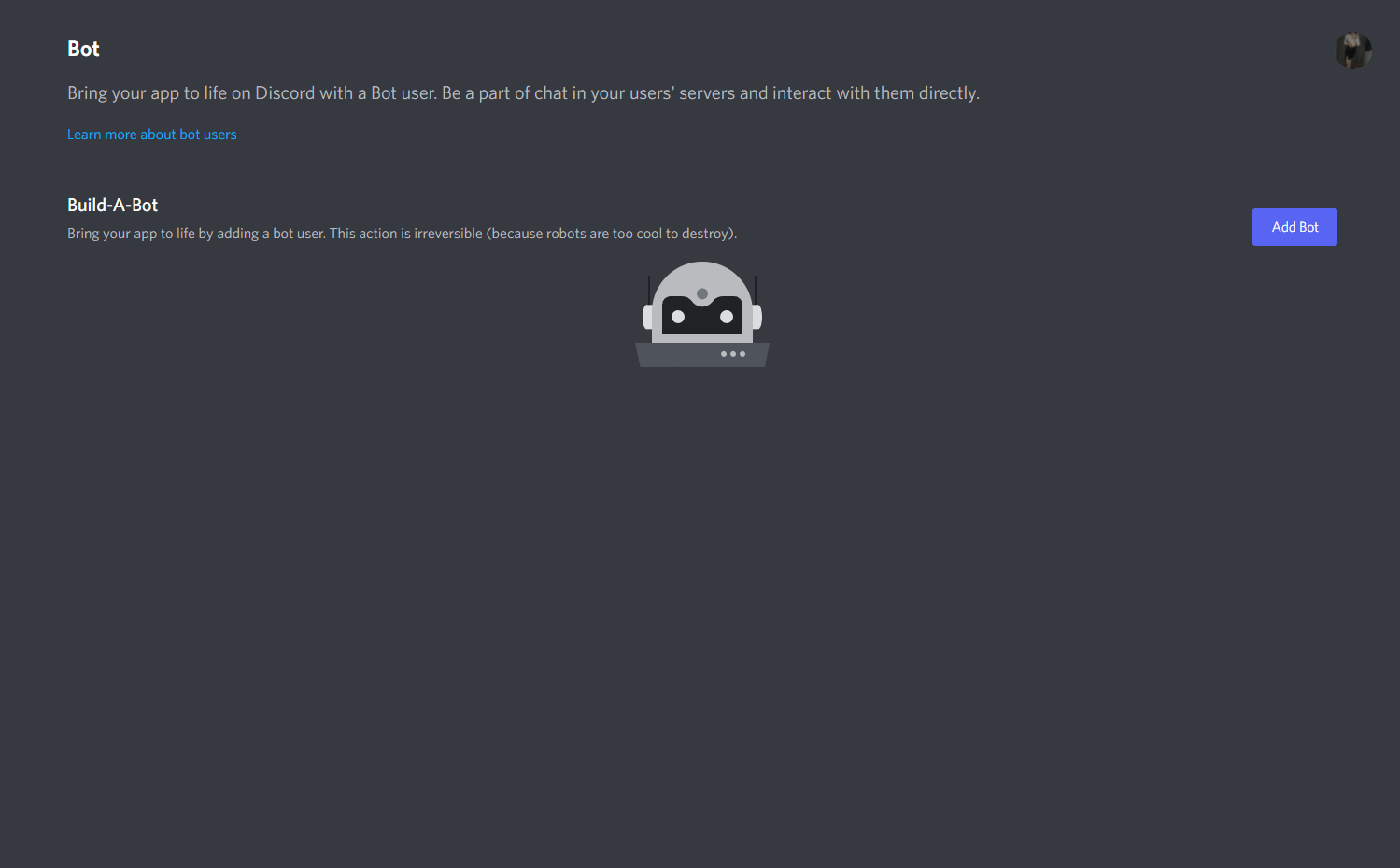


Рисунок 7 - Вкладка «Bot»

Появился бот, как показано на рисунке 8.

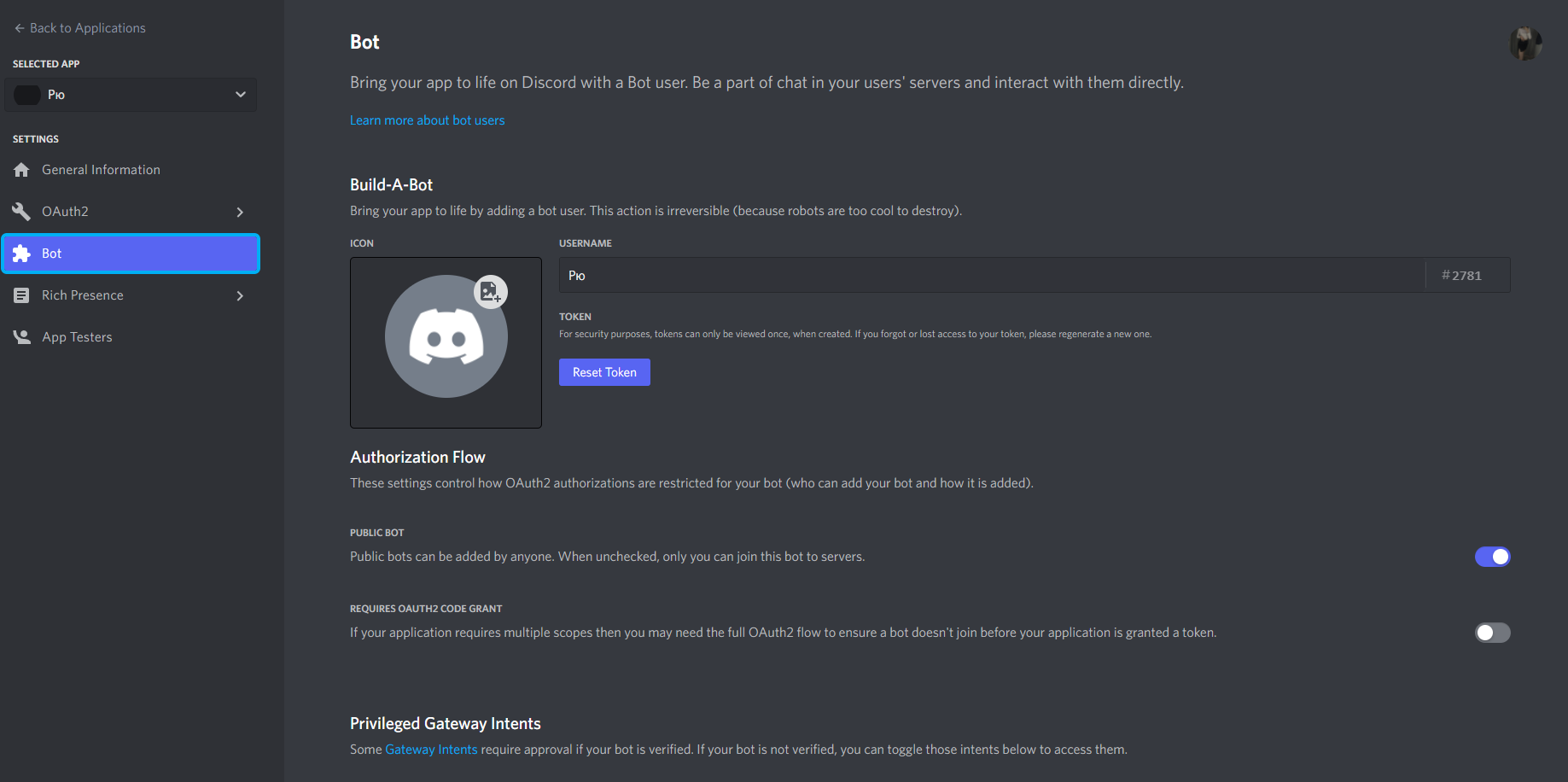


Рисунок 8 - Появление бота «Рю»

Также необходимо запомнить раздел токен и кнопку «Reset Token» для будущего запуска бота, как показано на рисунке 9.

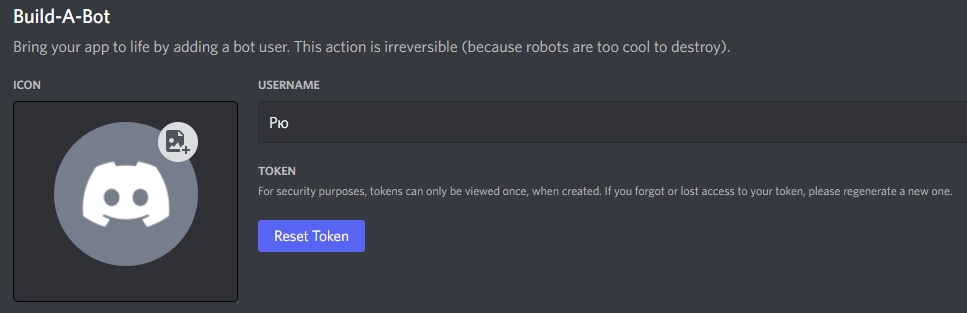


Рисунок 9 - Токен

Чтобы добавить бота «Рю» на сервер, нужно перейти во вкладку «OAuth2», в ней в раздел «URL Generator», в модуле «Scopes» выбрать «bot» и в модуле «Bot Permissions» выбрать «Administrator», как показано на рисунке 10.

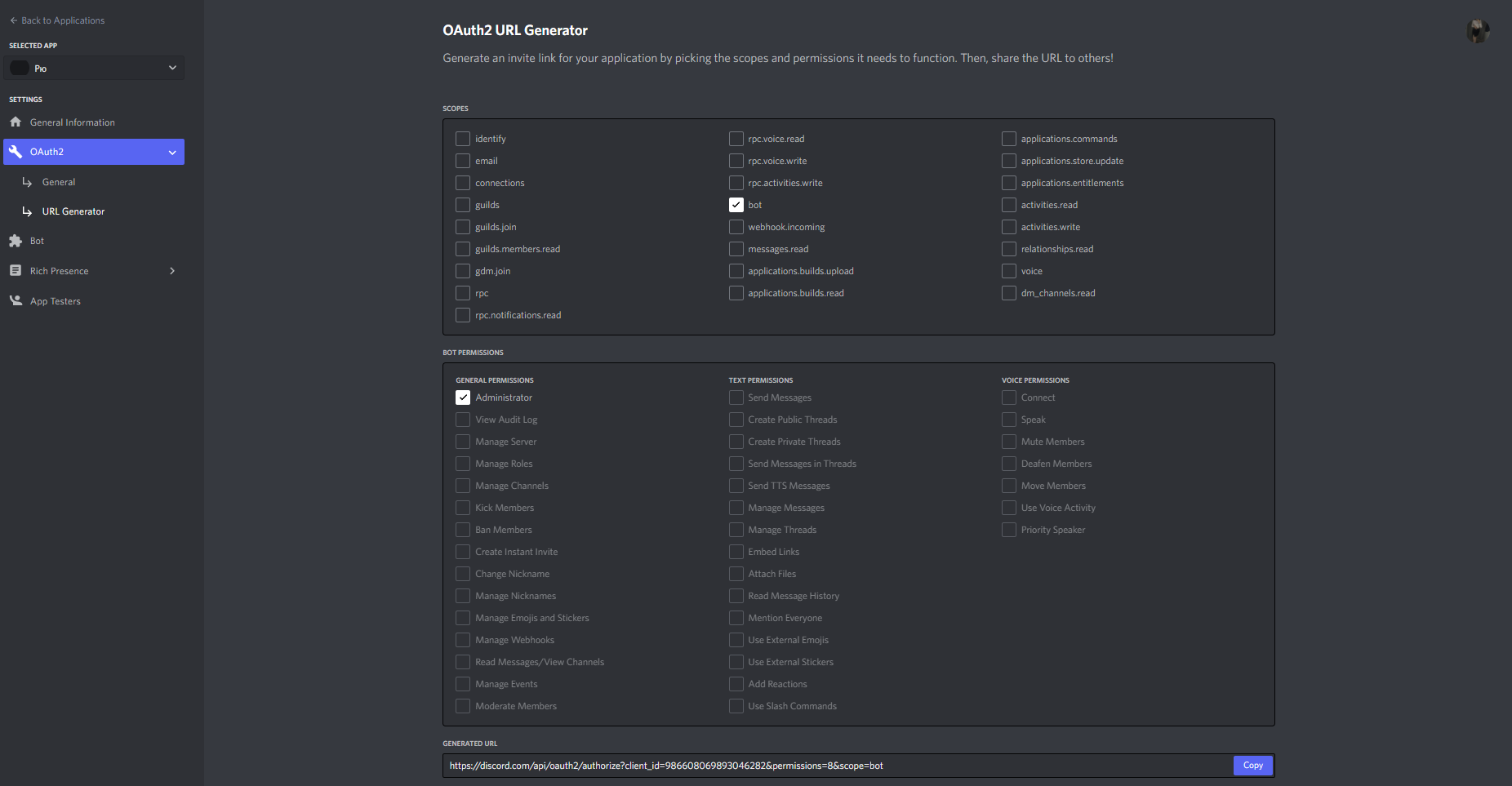


Рисунок 10 - Раздел «OAuth2»

Далее следует скопировать ссылку из раздела «Generated URL», вставить в браузер, после этого выбрать доступный сервер, нажать кнопку «Продолжить», а затем «Авторизовать», как показано на рисунках 11 и 12.

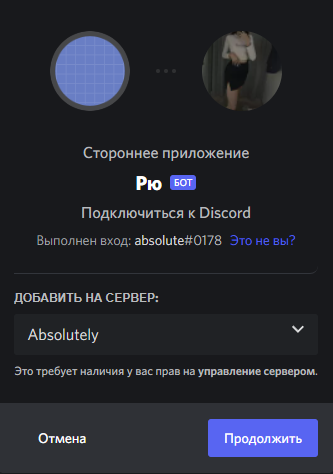


Рисунок 11 - Добавление бота на нужный сервер

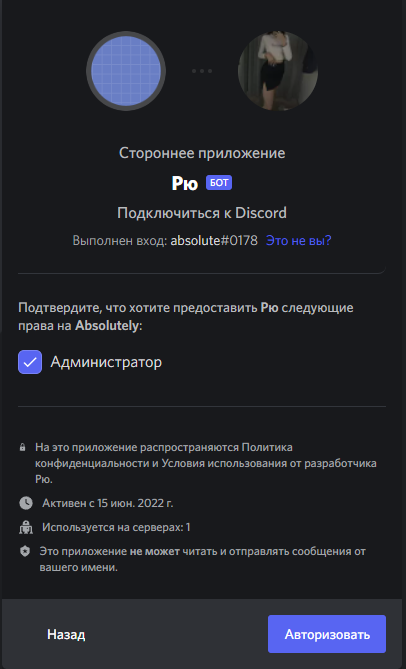


Рисунок 12 - Авторизация бота

При входе на сервер видно, что бот «Рю» появился в списке пользователей, как показано на рисунке 13.

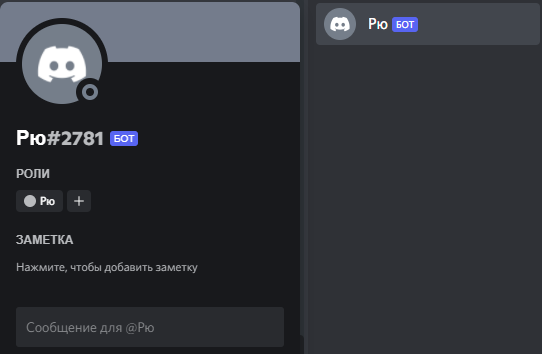


Рисунок 13 - Бот в списке пользователей

Далее нужно запустить бота в PyCharm с помощью токена, как показано на рисунке 14.



Рисунок 14 - Запуск бота из PyCharm

Бот запустился, вывелось соответствующее сообщение в консоль (Рисунок 15), а также бот появился в сети (Рисунок 16).



Рисунок 15 - Сообщение о запуске с полным именем бота «Рю»

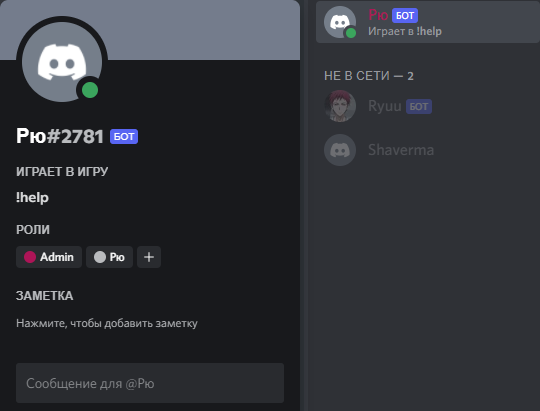


Рисунок 16 - Появление бота в сети

Необходимо проверить работоспособность бота. Для этого можно зайти в канал на сервере Discord и начать проверку, как изображено на рисунке 17.

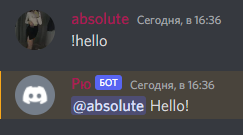


Рисунок 17 - Бот успешно запущен

Если пользователь использует неизвестную команду, то бот сообщит ему об этом, как показано на рисунке 18.

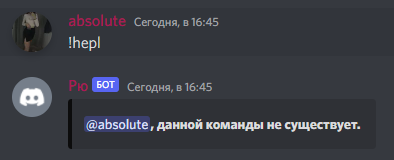


Рисунок 18 - Обработка неизвестных команд в чате

**Описание всех функций, выполняемых Discord ботом «Рю»:**

1. !help

После добавления бота на сервер, в информации об боте можно увидеть статус «Играет в !help», как показано на рисунке 19. Данная команда !help служит для изучения всех команд бота пользователем, как изображено на рисунке 20.

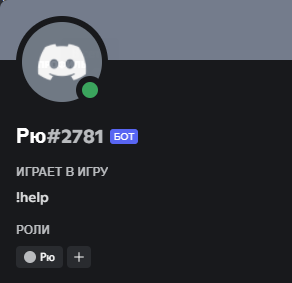


Рисунок 19 – Демонстрация статуса бота «Рю»



Рисунок 20 – Демонстрация работы команды !help

2) !hello

Бот «Рю» может поприветствовать пользователя при отправке данной команды в чат, как показано на рисунке 21.

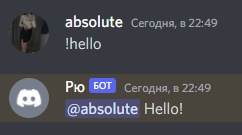


Рисунок 21 – Демонстрация работы команды !hello

3)!inspire

Если пользователь захочет получить фразу дзен, то он может обратиться к данной команде. Работа этой команды показана на рисунке 22.

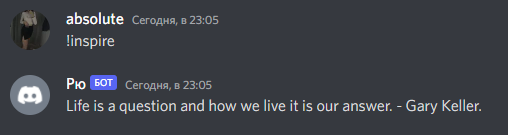


Рисунок 22 - Вывод команды !inspire

4)!clear

Данная команда нацелена только для работы администраторов сервера, с целью очистки чата от ненужного контента. Если этой функцией захочет воспользоваться тот, кто не имеет прав администратора, то в чат будет выдано оповещение, как показано на рисунке 23.

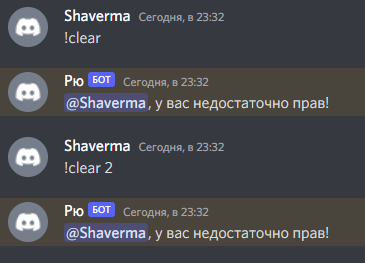


Рисунок 23 – Обработка отсутствия прав на использование команды !clear

Для этой команды также нужен аргумент, в случае его отсутствия выдастся оповещение, как показано на рисунке 24.

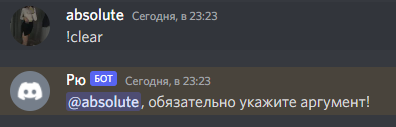


Рисунок 24 – Обработка отсутствия аргумента в команде !clear

В случае отправления в чат команды с аргументом, бот удалит количество сообщений, соизмеримое количеству, заданном в аргументе и выдаст оповещение об удалении этих сообщений, как показано на рисунке 25.

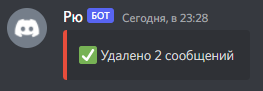


Рисунок 25 – Демонстрация работы команды !clear

5)!kick

Данная команда нацелена только для работы администраторов сервера, с целью удаления выбранного пользователя с сервера. Если этой функцией захочет воспользоваться тот, кто не имеет прав администратора, то в чат будет выдано оповещение, как показано на рисунке 26.

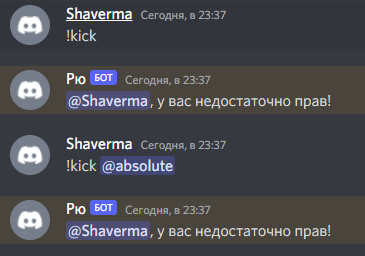


Рисунок 26 - Обработка отсутствия прав на использование команды !kick

Для этой команды также нужен аргумент, которым будет являться участник сервера, в случае отсутствия этого аргумента выдастся оповещение, как показано на рисунке 27.

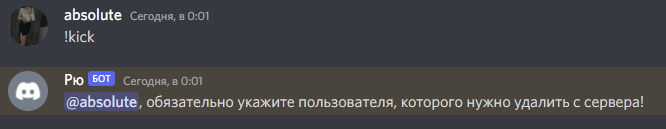


Рисунок 27 – Обработка отсутствия аргумента в команде !kick

В случае отправления в чат команды с аргументом, бот удалит выбранного пользователя с сервера и выдаст оповещение об этом в чат, как показано на рисунке 28.

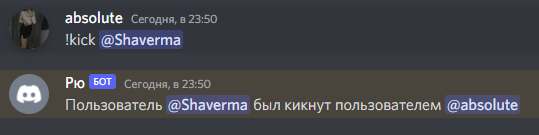


Рисунок 28 – Демонстрация работы команды !kick

6)!ban

Данная команда нацелена только для работы администраторов сервера, с целью ограничения доступа пользователя к серверу. Если этой функцией захочет воспользоваться тот, кто не имеет прав администратора, то в чат будет выдано оповещение, как показано на рисунке 29.

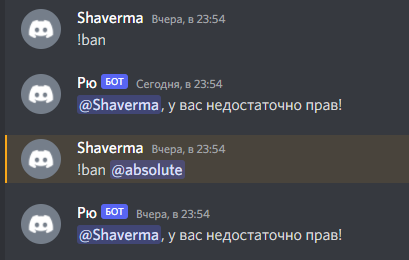


Рисунок 29 – Обработка отсутствия прав на использование команды !ban

Для этой команды также нужен аргумент, которым будет являться участник сервера, в случае отсутствия этого аргумента выдастся оповещение, как показано на рисунке 30.

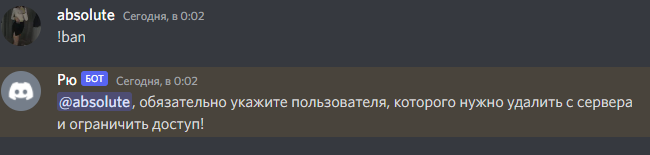


Рисунок 30 – Обработка отсутствия аргумента в команде !ban

В случае отправления в чат команды с аргументом, бот удалит и ограничит доступ к серверу у выбранного пользователя, а также выдаст оповещение об этом, как показано на рисунке 31.

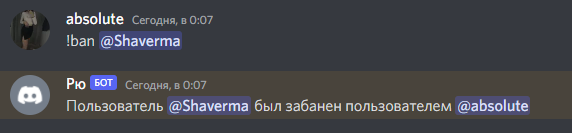


Рисунок 31 – Демонстрация работы команды !ban

7)!mute

Данная команда нацелена только для работы администраторов сервера, с целью ограничения пользования чатом у выбранного пользователя. Если этой функцией захочет воспользоваться тот, кто не имеет прав администратора, то в чат будет выдано оповещение, как показано на рисунке 32.

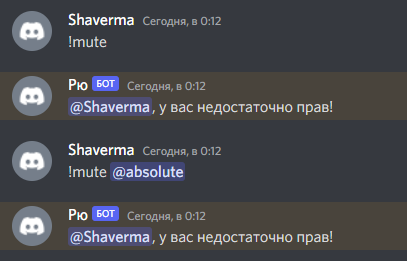


Рисунок 32 – Обработка отсутствия прав на использование команды !mute

Для этой команды также нужен аргумент, которым будет являться участник сервера, в случае отсутствия этого аргумента выдастся оповещение, как показано на рисунке 33.

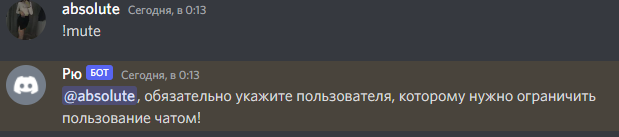


Рисунок 33 – Обработка отсутствия аргумента в команде !mute

В случае отправления в чат команды с аргументом, бот выдаст выбранному участнику роль «mute», что изображено на рисунке 34, следовательно, он не сможет обмениваться сообщениями в чате, как показано на рисунке 35, а также выдаст оповещение об ограничении пользования чатом. Данная операция описана на рисунке 36.

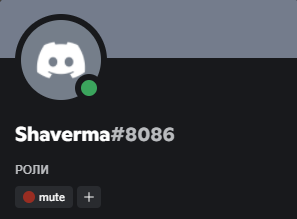


Рисунок 34 – Выдача участнику сервера роли «mute»

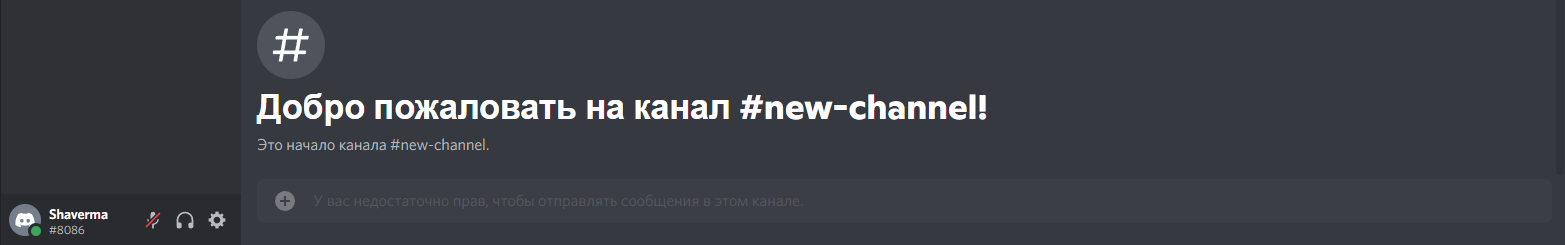


Рисунок 35 – Демонстрация ограничения чата пользователю с ролью «mute»

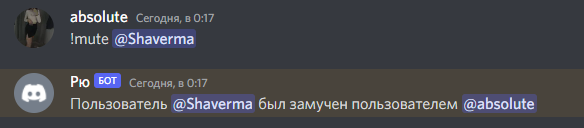


Рисунок 36 – Демонстрация работы команды !mute

8)!unmute

Данная команда нацелена только для работы администраторов сервера, с целью отмены ограничения пользования чатом. Если этой функцией захочет воспользоваться тот, кто не имеет прав администратора, то в чат будет выдано оповещение, как показано на рисунке 37.

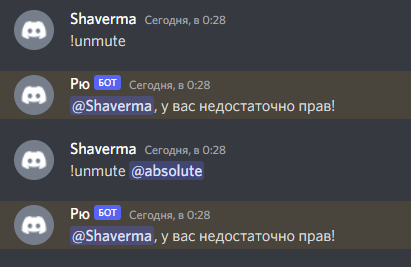


Рисунок 37 – Обработка отсутствия прав на использование команды !unmute

Для этой команды также нужен аргумент, которым будет являться участник сервера, в случае отсутствия этого аргумента выдастся оповещение, как показано на рисунке 38.

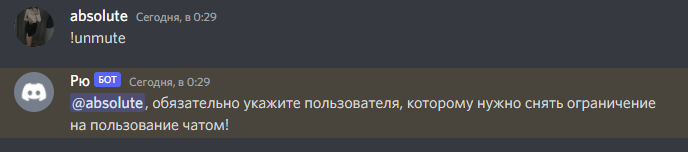


Рисунок 38 – Обработка отсутствия аргумента в команде !unmute

В случае отправления в чат команды с аргументом, бот снимет с выбранного участнику роль «mute», как показано на рисунке 39, следовательно, он сможет обмениваться сообщениями в чате, а также выдаст оповещение об отмене ограничения на использование чата. Данная операция описана на рисунке 40.

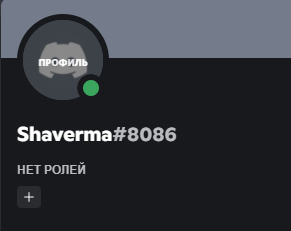


Рисунок 39 – Снятие роли «mute» с пользователя

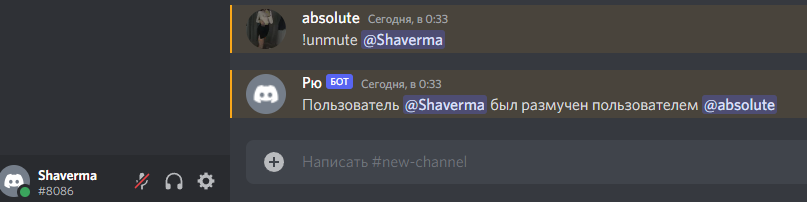


Рисунок 40 – Демонстрация работы команды !unmute

9) !time

Данная команда выводит реальное время в чат сервера.

При вызове в чате выводится:

* Заголовок «Время»
* Описание данной команды
* Изображение с часами
* Точное время
* Аватар и имя пользователя, который использовал данную команду

Описание этой команды изображено на рисунке 41.

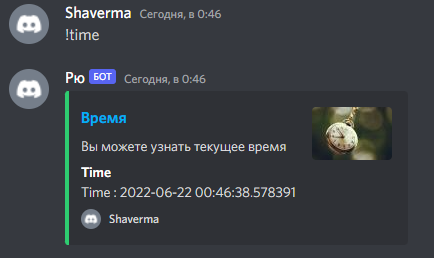


Рисунок 41 – Демонстрация работы команды !time

10) !gratitude

Данная команда посылает благодарность за пользование ботом «Рю» напрямую в личные сообщения пользователя, который вызвал эту функцию, как показано на рисунке 42.

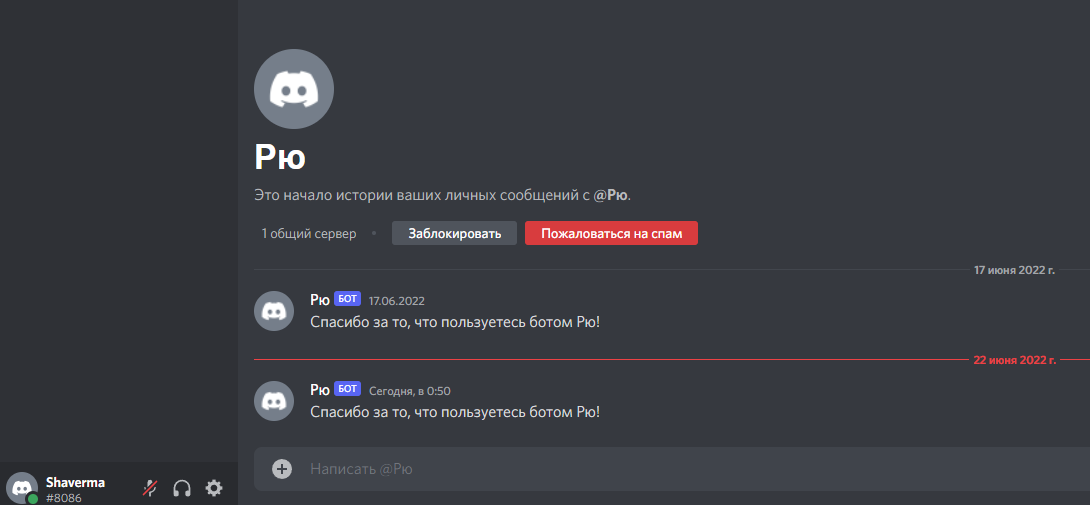


Рисунок 42 – Демонстрация работы команды !gratitude

11) !join

Для использования данной команды пользователю нужно находится в каком-либо голосовом канале сервера. При вызове этой функции бот перемещается к участнику в голосовой канал, что описано на рисунке 43, а также оповещает, в какой канал он переместился, как показано на рисунке 44.

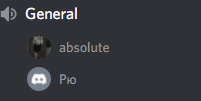


Рисунок 43 – Перемещение бота к пользователю в голосовой канал

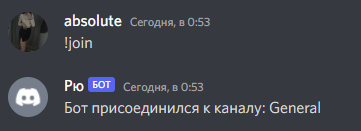


Рисунок 44 – Оповещение бота о названии канала, в который он переместился

12) !leave

Для использования данной команды пользователю и боту нужно находится в каком-либо голосовом канале сервера. При вызове этой функции бот выходит из голосового канала, как показано на рисунке 45.

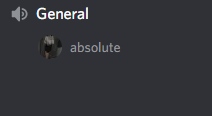


Рисунок 45 – Выход бота из голосового канала, используя команду !leave

13) Вывод ободряющих сообщений

Если любой пользователь в чате напишет о том, что ему грустно или тоскливо, то бот поддержит его, выдав ободряющее сообщение в чат сервера, как показано на рисунке 46.

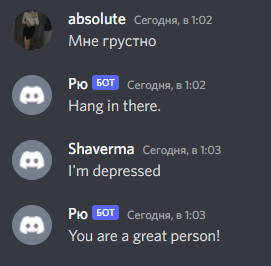


Рисунок 46 – Демонстрация работы вывода ободряющих сообщений

**Заключение.**

По итогу разработки в курсовом проекте было разработано:

* Приветствие пользователя;
* Генератор цитат с помощью API и HTTP-запросов;
* Очистка чата;
* Удаление участника с сервера;
* Ограничение доступа к серверу;
* Вывод реального времени;
* Ограничение пользования чатом;
* Отмена ограничения пользования чатом;
* Отправление сообщения ботом напрямую в личные сообщения пользователя;
* Подключение к голосовому каналу;
* Отключение от голосового канала.

Изначальные цели и задачи проекта выполнены.

**Список использованной литературы и интернет - ресурсов.**

## Законодательные и нормативные акты:

1. ГОСТ 7.1. – 2003. Библиографическая запись. Библиографическое описание. Общие требования и правила составления. – М.: ИПК Издательство стандартов, 2004. – 169 с.
2. ГОСТ 7.32 – 2017 Система стандартов по информации, библиотечному и издательскому делу (с поправками) – М.: ИПК Издательство стандартов, 2017. – 21 с.
3. Единая система программной документации. – М.: Стандартинформ, 2005. – 128 с.

## Учебная и научная литература:

1. Любанович Билл. Простой Python. Современный стиль программирования. — СПб.: Питер, 2016. — 480 с.

## Интернет-документы:

1. Stack Overflow– [Электронный ресурс]. – Режим доступа: <https://stackoverflow.com/>
2. Русский Stack Overflow – [Электронный ресурс]. – Режим доступа: <https://ru.stackoverflow.com/>
3. Discord API Reference - <https://discordpy.readthedocs.io/en/stable/api.html>;
4. Stack Overflow - <https://stackoverflow.com/>;
5. GitHub - <https://github.com>.

**Приложения.**

**Приложение 1. Файл main.py**

import os  
from discord.ext import commands  
from settings\_local import token  
  
client = commands.Bot(command\_prefix='!')  
client.remove\_command('help')  
  
for filename in os.listdir('./Discord\_Bot'):  
 if filename.endswith('.py'):  
 client.load\_extension(f'Discord\_Bot.{filename[:-3]}')  
  
client.run(token)

**Приложение 2. Файл Events.py**

import discord  
import random  
from discord.ext import commands  
from Discord\_Bot.List\_words.list\_words import sad\_words, words\_encouragement  
  
class Events(commands.Cog):  
 def \_\_init\_\_(self, client):  
 self.client = client  
  
 @commands.Cog.listener()  
 async def on\_ready(self):  
 """Вывод статуса бота в консоль при подключении на сервер"""  
 print('Бот подключен под именем {0.user}'.format(self.client))  
  
 await self.client.change\_presence(status=discord.Status.online, activity=discord.Game('!help'))  
  
 @commands.Cog.listener()  
 async def on\_message(self, message):  
 """Вывод ободряющих фраз для пользователя в случае отправки грустного слова"""  
 msg = message.content.lower()  
  
 if any(word in msg for word in sad\_words):  
 await message.channel.send(random.choice(words\_encouragement))  
  
 @commands.Cog.listener()  
 async def on\_command\_error(self, ctx, error):

**Приложение 2.**

"""Вывод ошибки о неизвестной команде в чат"""  
 if isinstance(error, commands.CommandNotFound):  
 await ctx.send(embed = discord.Embed(description = f'{ctx.author.mention}, данной команды не существует.',  
 color=0x0c0c0c))  
  
def setup(client):  
 client.add\_cog(Events(client))

**Приложение 3. Файл User.py**

import discord  
from discord.ext import commands  
import requests  
import json  
import datetime  
from discord.utils import get  
  
class User(commands.Cog):  
 def \_\_init\_\_(self, client: commands.Bot):  
 self.client = client  
  
 @commands.command()  
 async def inspire(self, ctx):  
 """Команда inspire, которая выводит полученные от функции get\_quote цитаты"""  
 def get\_quote():  
 """Функция для получения цитат с помощью API и HTTP-запроса"""  
 response = requests.get("https://zenquotes.io/api/random")  
  
 json\_data = json.loads(response.text)  
 quote = json\_data[0]['q'] + ' - ' + json\_data[0]['a'] + '.'  
 return quote  
  
 quote = get\_quote()  
 await ctx.send(quote)  
  
 @commands.command()  
 async def hello(self, ctx):

**Приложение 3.**

"""Команда hello"""  
 author = ctx.message.author  
  
 await ctx.send(f'{author.mention}' + ' Hello!')  
  
 @commands.command()  
 async def help(self, ctx):  
 """Команда help"""  
 emb = discord.Embed(title = 'Навигация по командам')  
  
 emb.add\_field(name='{}hello'.format('!'), value='Поздоровайтесь с ботой')  
 emb.add\_field(name='{}inspire'.format('!'),  
 value='Цитаты, которые принесут спокойствие в вашу жизнь')  
 emb.add\_field(name='{}clear'.format('!'), value = 'Очистка чата\n(только для администраторов)')  
 emb.add\_field(name='{}kick'.format('!'), value='Удаление участника с сервера\n(только для администраторов)')  
 emb.add\_field(name='{}ban'.format('!'), value='Ограничение доступа к серверу\n(только для администраторов)')  
 emb.add\_field(name='{}time'.format('!'), value='Вывод реального времени')  
 emb.add\_field(name='{}mute'.format('!'), value='Ограничение пользования чатом\n(только для администраторов)')  
 emb.add\_field(name='{}unmute'.format('!'),  
 value='Отмена ограничения пользования чатом\n(только для администраторов)')  
 emb.add\_field(name='{}gratitude'.format('!'),  
 value='Благодарность за пользование в личном сообщении')  
 emb.add\_field(name='{}join'.format('!'), value='Подключение к голосовому чату')

**Приложение 3.**

emb.add\_field(name='{}leave'.format('!'), value='Отключение от голосового чата')  
 emb.add\_field(name='Вывод ободряющих сообщений',  
 value='Если вам станет грустно, поделитесь этим, и бот вас обязательно поддержит')  
  
 await ctx.send(embed = emb)  
  
 @commands.command()  
 async def gratitude(self, ctx):  
 """Команда для отправки сообщения от бота в лс пользователя"""  
 await ctx.author.send('Спасибо за то, что пользуетесь ботом Рю!')  
  
 @commands.command()  
 async def time(self, ctx):  
 """Команда time"""  
 emb = discord.Embed(title='Время', description='Вы можете узнать текущее время',  
 colour=discord.Color.green(), url='https://time100.ru/')  
 emb.set\_footer(text=ctx.author.name, icon\_url=ctx.author.avatar\_url)  
 emb.set\_thumbnail(url='https://i.pinimg.com/originals/8e/04/d6/8e04d62d3c4d1c9e514d2eb99a32c893.jpg')  
  
 now\_date = datetime.datetime.now()  
 emb.add\_field(name='Time', value='Time : {}'.format(now\_date))  
 await ctx.send(embed=emb)  
  
 @commands.command()

**Приложение 3.**

async def join(self, ctx):  
 """Команда join - подключение бота к голосовому чату"""  
 channel = ctx.message.author.voice.channel  
 voice = get(self.client.voice\_clients, guild=ctx.guild)  
  
 if voice and voice.is\_connected():  
 await voice.move\_to(channel)  
 else:  
 voice = await channel.connect()  
 await ctx.send(f'Бот присоединился к каналу: {channel}')  
  
 @commands.command()  
 async def leave(self, ctx):  
 """Команда leave - отключение от голосового чата"""  
 channel = ctx.message.author.voice.channel  
 voice = get(self.client.voice\_clients, guild=ctx.guild)  
  
 if voice and voice.is\_connected():  
 await voice.disconnect()  
 else:  
 voice = await channel.connect()  
  
def setup(client: commands.Bot):  
 client.add\_cog(User(client))

**Приложение 4. Файл Admin.py**

import discord  
from discord.ext import commands  
  
class Admin(commands.Cog):  
 @commands.command()  
 @commands.has\_permissions(administrator = True)  
 async def clear(self, ctx, amount : int):  
 """Команда clear - очистка чата"""  
 await ctx.channel.purge(limit = (amount+1))  
  
 await ctx.send(embed = discord.Embed(description=f':white\_check\_mark: Удалено {amount} сообщений',  
 color=discord.Color.red()))  
  
 @clear.error  
 async def clear\_error(self, ctx, error):  
 """Обработка ошибок у команды clear"""  
 if isinstance(error, commands.MissingRequiredArgument):  
 await ctx.send(f'{ctx.author.mention}, обязательно укажите аргумент!')  
  
 if isinstance(error, commands.MissingPermissions):  
 await ctx.send(f'{ctx.author.mention}, у вас недостаточно прав!')  
  
 @commands.command()  
 @commands.has\_permissions(administrator = True)  
 async def kick(self, ctx, member: discord.Member, \*, reason = None):  
 """Команда kick - удаление участника с сервера"""  
 await member.kick(reason = reason)

**Приложение 4.**

await ctx.send(f'Пользователь {member.mention} был кикнут пользователем {ctx.author.mention}')  
  
 @kick.error  
 async def kick\_error(self, ctx, error):  
 """Обработка ошибок у команды kick"""  
 if isinstance(error, commands.MissingPermissions):  
 await ctx.send(f'{ctx.author.mention}, у вас недостаточно прав!')  
  
 if isinstance(error, commands.MissingRequiredArgument):  
 await ctx.send(f'{ctx.author.mention}, обязательно укажите пользователя, которого нужно удалить с сервера!')  
  
 @commands.command()  
 @commands.has\_permissions(administrator = True)  
 async def ban(self, ctx, member: discord.Member, \*, reason = None):  
 """Команда ban - ограничение доступа к серверу"""  
 await member.ban(reason = reason)  
 await ctx.send(f'Пользователь {member.mention} был забанен пользователем {ctx.author.mention}')  
  
 @ban.error  
 async def ban\_error(self, ctx, error):  
 """Обработка ошибок у команды ban"""  
 if isinstance(error, commands.MissingPermissions):  
 await ctx.send(f'{ctx.author.mention}, у вас недостаточно прав!')  
  
 if isinstance(error, commands.MissingRequiredArgument):

**Приложение 4.**

await ctx.send(f'{ctx.author.mention}, обязательно укажите пользователя, которого нужно удалить с сервера '  
 f'и ограничить доступ!')  
  
 @commands.command()  
 @commands.has\_permissions(administrator = True)  
 async def mute(self, ctx, member: discord.Member):  
 """Команда mute - ограничение пользования чатом"""  
 mute\_role = discord.utils.get(ctx.message.guild.roles, name = 'mute')  
  
 await member.add\_roles(mute\_role)  
 await ctx.send(f'Пользователь {member.mention} был замучен пользователем {ctx.author.mention}')  
  
 @mute.error  
 async def mute\_error(self, ctx, error):  
 """Обработка ошибок у команды mute"""  
 if isinstance(error, commands.MissingPermissions):  
 await ctx.send(f'{ctx.author.mention}, у вас недостаточно прав!')  
  
 if isinstance(error, commands.MissingRequiredArgument):  
 await ctx.send(f'{ctx.author.mention}, обязательно укажите пользователя, которому нужно ограничить '  
 f'пользование чатом!')  
  
 @commands.command()  
 @commands.has\_permissions(administrator = True)  
 async def unmute(self, ctx, member: discord.Member):

**Приложение 4.**

"""Команда unmute - отмена ограничения пользования чатом"""  
 mute\_role = discord.utils.get(ctx.message.guild.roles, name = 'mute')  
  
 await member.remove\_roles(mute\_role)  
 await ctx.send(f'Пользователь {member.mention} был размучен пользователем {ctx.author.mention}')  
  
 @unmute.error  
 async def unmute\_error(self, ctx, error):  
 """Обработка ошибок у команды unmute"""  
 if isinstance(error, commands.MissingPermissions):  
 await ctx.send(f'{ctx.author.mention}, у вас недостаточно прав!')  
  
 if isinstance(error, commands.MissingRequiredArgument):  
 await ctx.send(f'{ctx.author.mention}, обязательно укажите пользователя, которому нужно снять ограничение '  
 f'на пользование чатом!')  
  
def setup(client: commands.Bot):  
 client.add\_cog(Admin(client))

**Приложение 5. Файл settings.py**

token = 'bad token'  
try:  
 from settings\_local import \*  
except:  
 pass

**Приложение 6. Файл list\_words.py**

"""Два списка слов для вывода ободряющих фраз"""  
sad\_words = ["sad", "depressed", "unhappy", "angry",  
 "грусть", "печаль", "тоска", "злость", "депрессия",  
 "грустно", "печально", "тоскливо"]  
  
words\_encouragement = [  
 "Cheer up!",  
 "Hang in there.",  
 "You are a great person!",  
 "Не опускай руки!",  
 "Ты отличный человек!",  
 "Не вешай нос, все будет в порядке"  
]