NYP LIT

# Discord Bot Workshop

By Planning Committee

# Contents

Pre-Workshop Preparations
Getting Started
Explanation of how the discord bot work
Creation of discord bot
Coding hands-on
Kahoot Time
Demonstration of final product

# Pre-Workshop Preparations

# Pre-Workshop Preparations

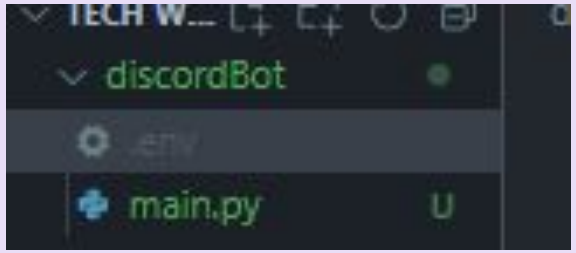
Make sure you have these prepared before the workshop:

1. Discord Developer Portal https://discord.com/developers/docs/resources/channel (can be accessed by signing in, there is no need to download).
2. Python and PyCharm/VSCode.
3. Download Git Heroku CLI
4. An account in Heroku.
5. A new server/your own server to test the bot on.

Note! You may refer to the PDF if you are unsure how to install these or reach out to our LIT members for help via Discord.

# Let’s get started!

# Setting up




1. Create Virtual Environment (optional)
2. Create main.py file
3. Create .env file

# Setting up

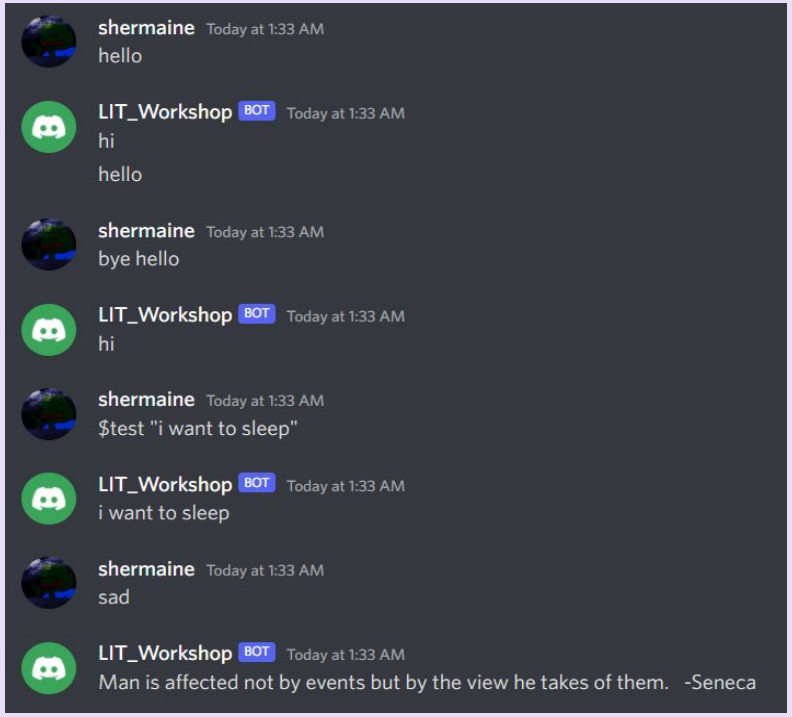
4. Run the following commands in your terminal/command prompt.
    - pip install python-dotenv

5. In your .env file, copy the following and fill in the empty single quotes with your own Discord token).
    - DISCORD_TOKEN = ‘{your token}‘

Note! Do not share your DISCORD_TOKEN with others! It should be kept secret!

# Discord Bot in Action

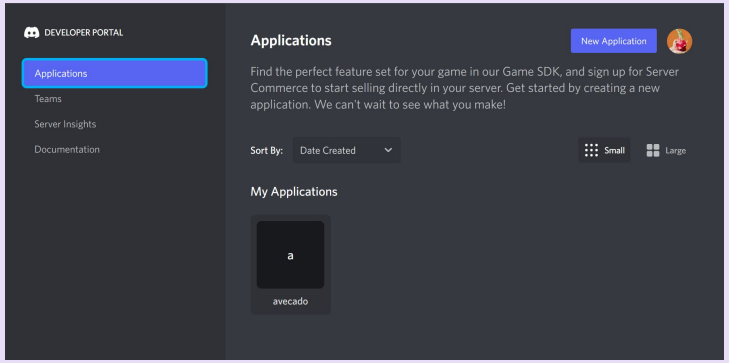
- The Bot replies to “Hello” message from user with “Hi”.
- When the word “sad” is sent by the user, the Bot will reply with a randomly selected quote from an API.



# Creation of Discord Bot

# How does the Discord Bot work?

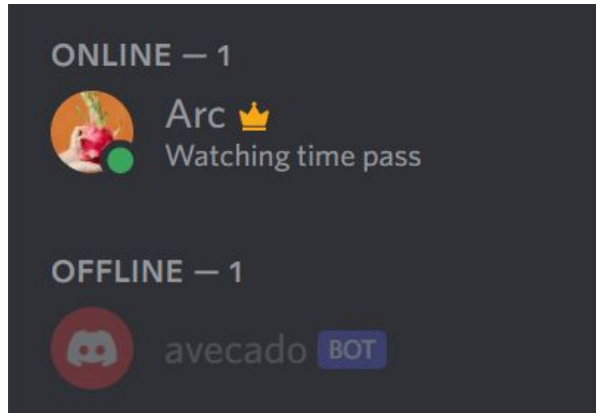
- User creates an application in the discord developer portal.
- User writes the code for the Bot in a text editor (e.g PyCharm).
- **When the code runs, the Bot created will run in the server that they sent the bot into on discord.**




```
const Discord = require('discord.js');
const client = new Discord.Client();

client.once('ready', () => {
    console.log('Ready!');
});

client.login('---');

client.on('message', message => {
    console.log(message.content);
});

if (message.content === '!Spaghetti') {
    message.channel.send('Bolognase !');
}
```

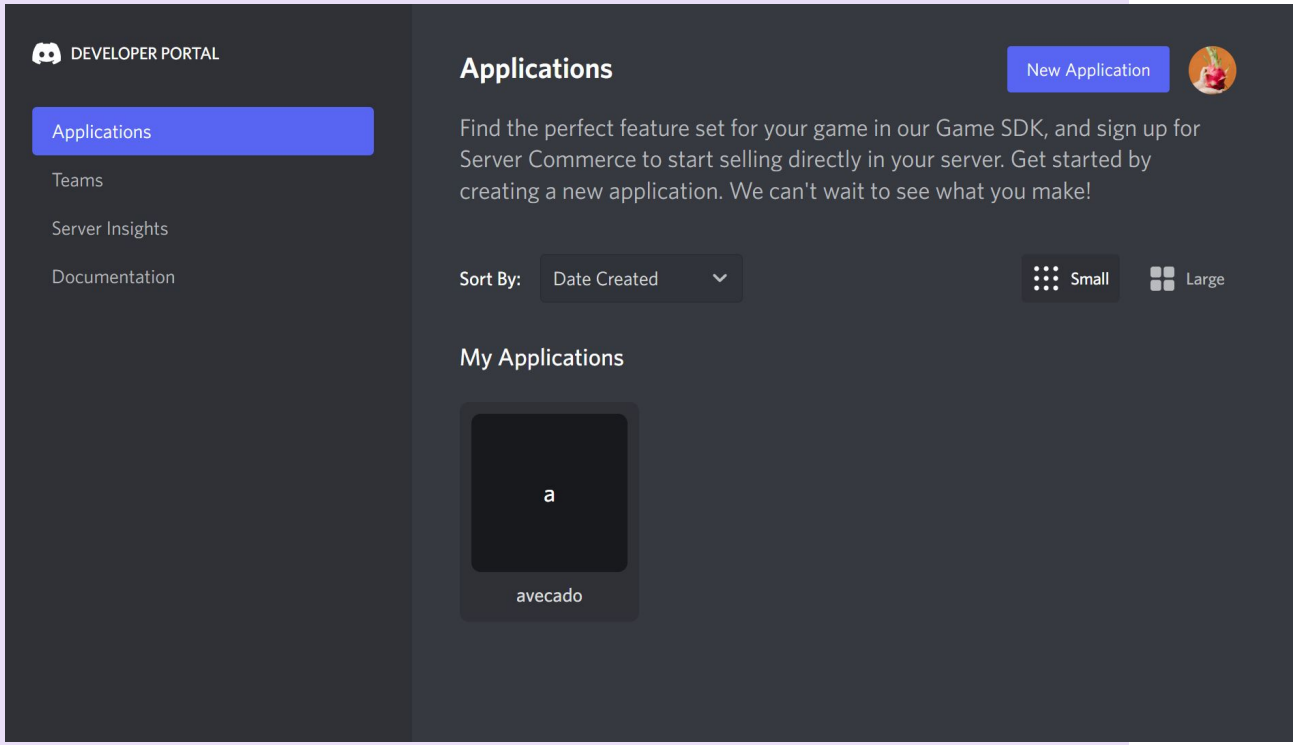
# How does the Discord Bot work?

- **The bot and discord will only be live when the code is run.**
- **An alternative is by hosting it online on platforms like heroku.**
- The User's personal computer runs both the Discord Client and the code so that the bot is able to send and read messages coming in on Discord.



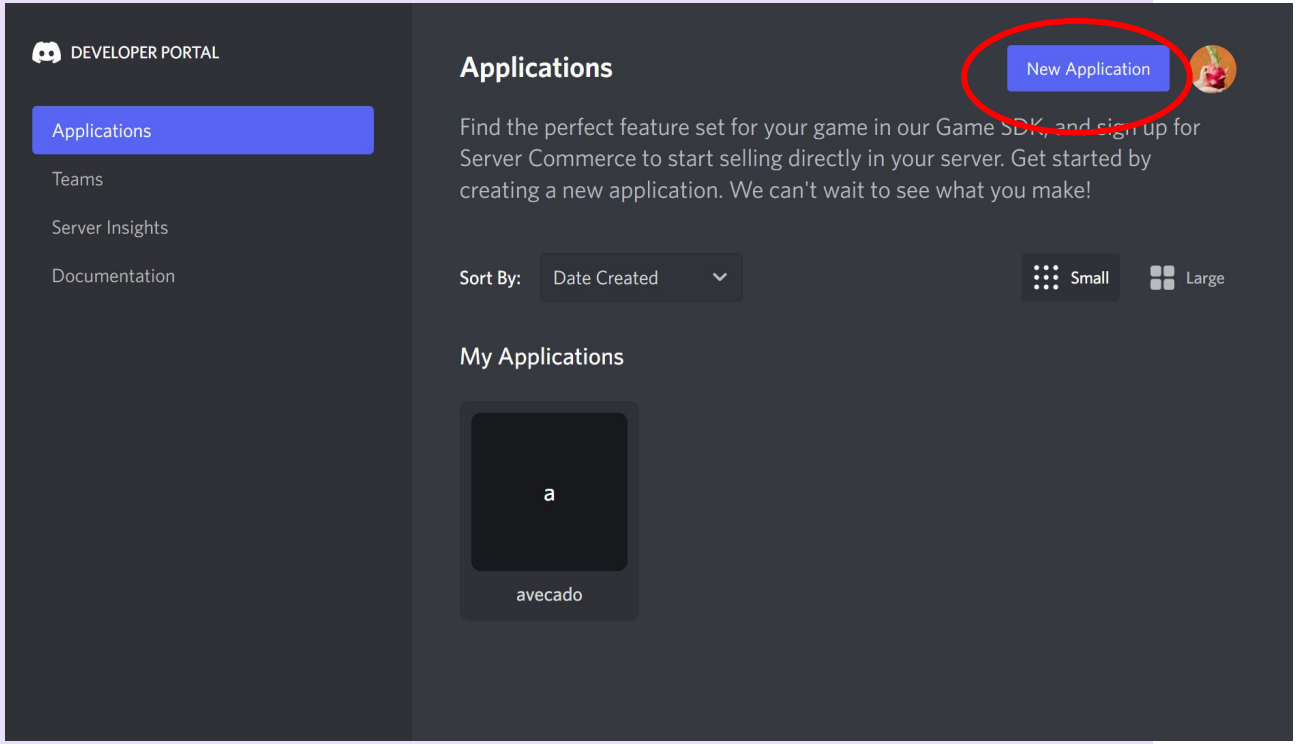
# Creating a bot account

Start off by visiting the following link: https://discord.com/developers/applications/



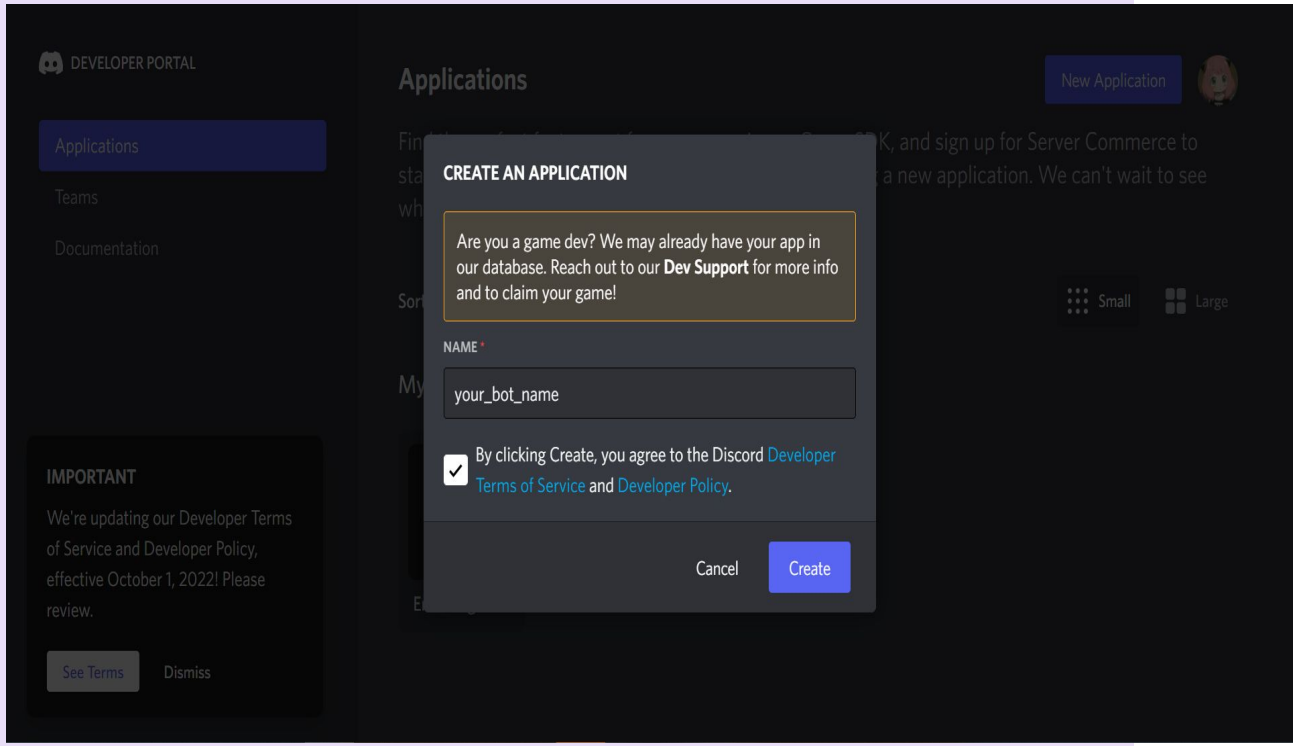
# Creating a bot account

Click “New Application” in the top right corner, name your application and click create.



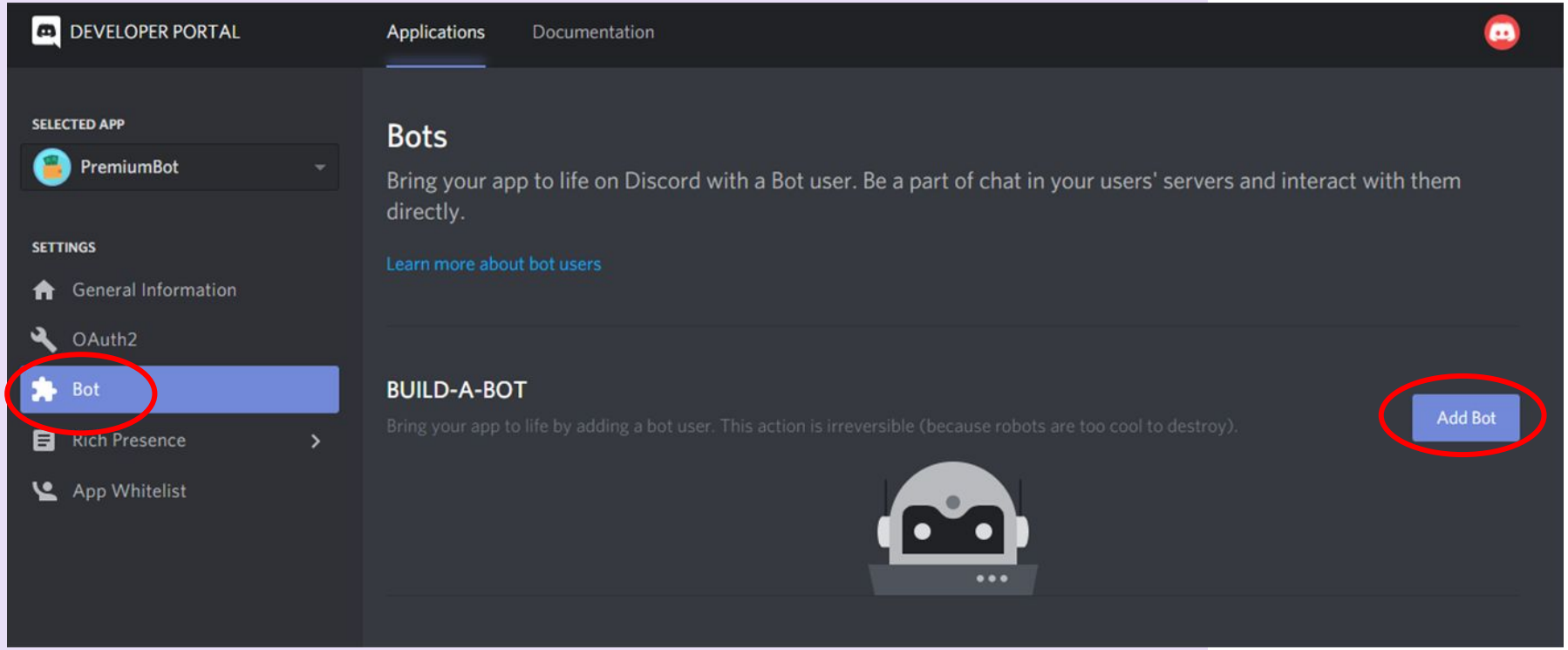
# Creating a bot account

Click “New Application” in the top right corner, name your application and click create.



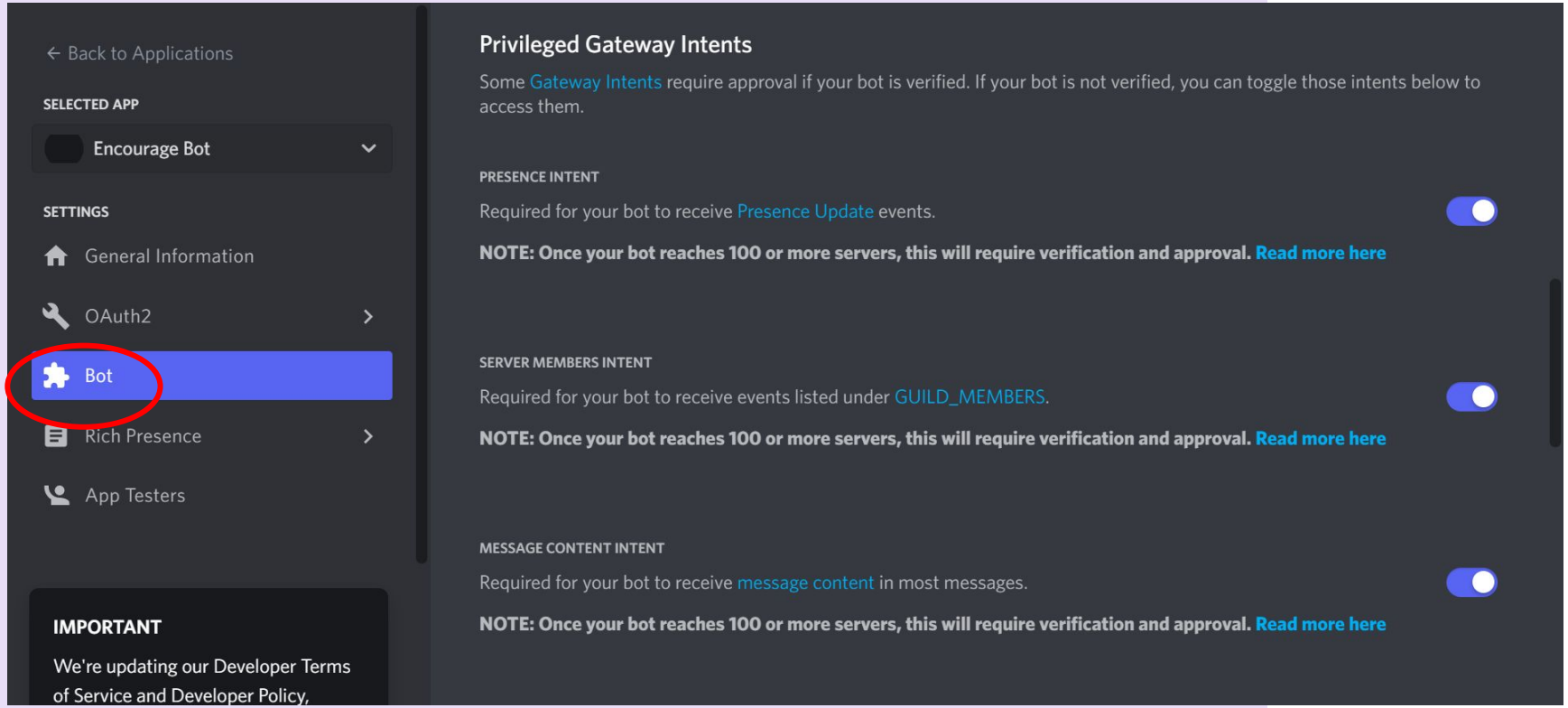
# Creating a bot account

Go to the Bot section and select “Add Bot”.



# Creating a bot account

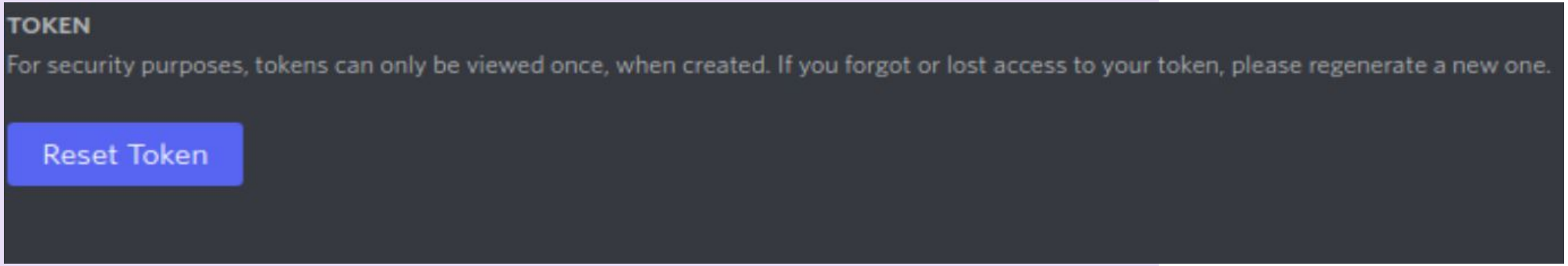
Scroll down and enable Privileged Gateway Intents.



# Creating a bot account

Here, we can obtain the discord token for our bot.

This token is what helps to uniquely identify our bot, and authenticate that we have the necessary permissions for the bot.

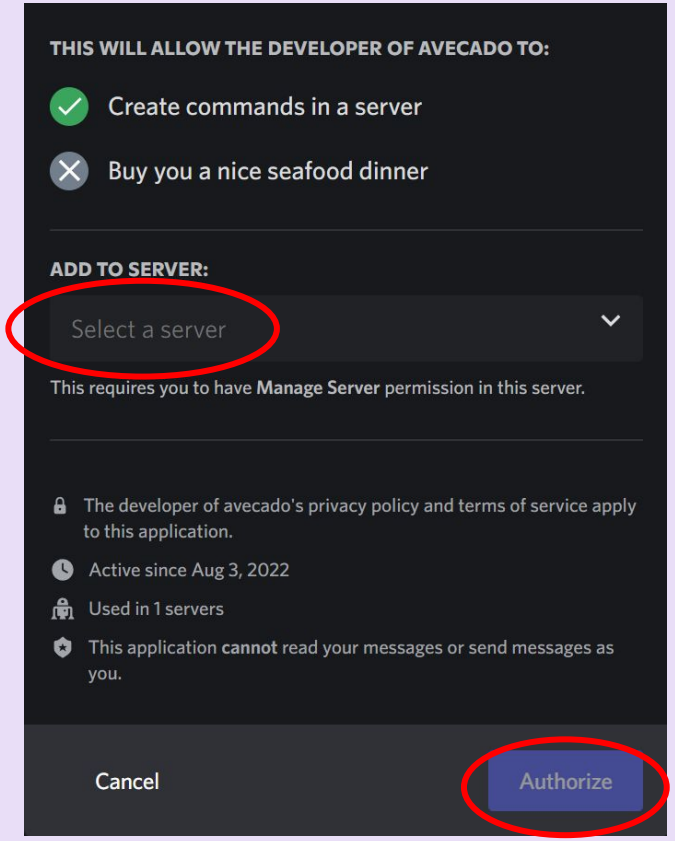


# Creating a bot account

After setting permissions, copy generated URL into your browser.

# Creating a bot account

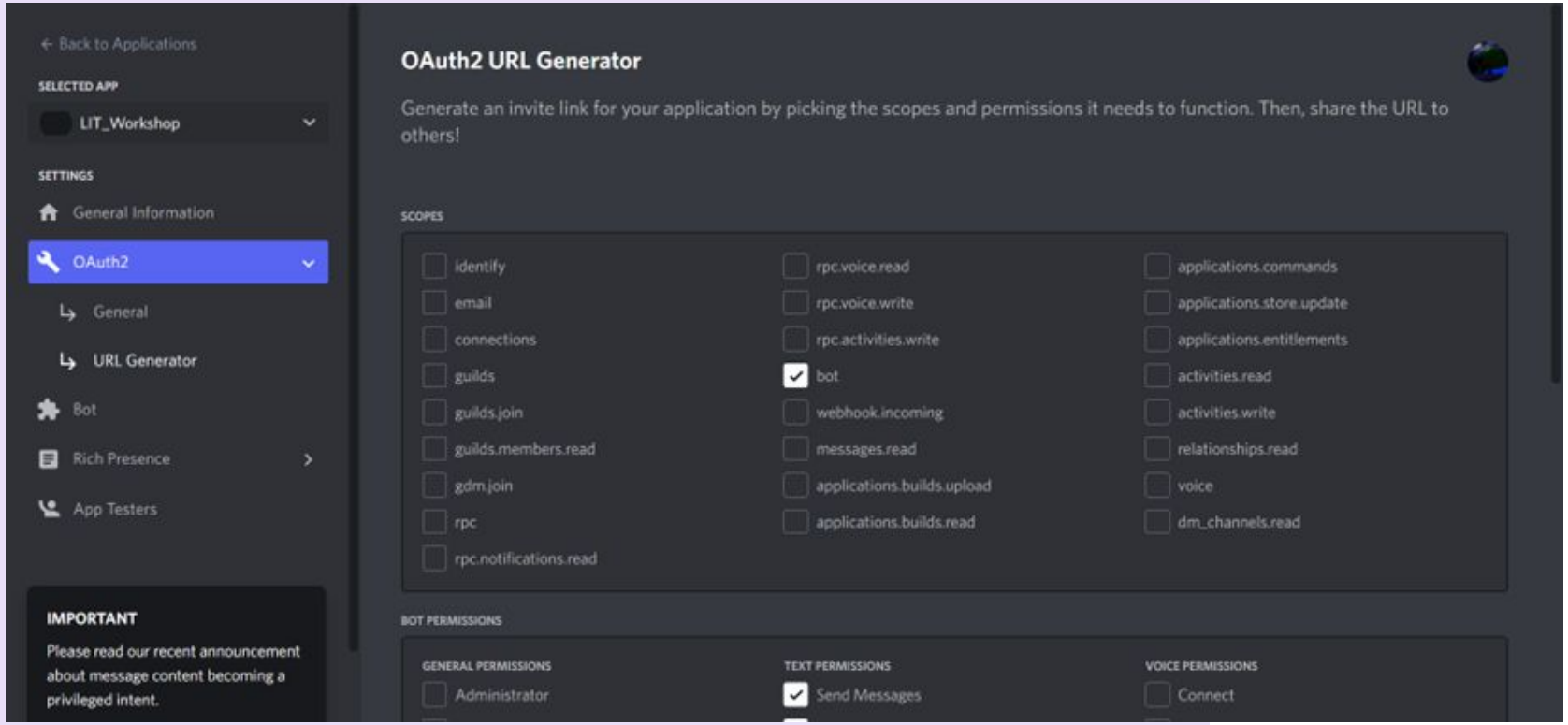
Select a test server you intend to invite your bot into and select “Authorize”.



# Sending Bot to Server

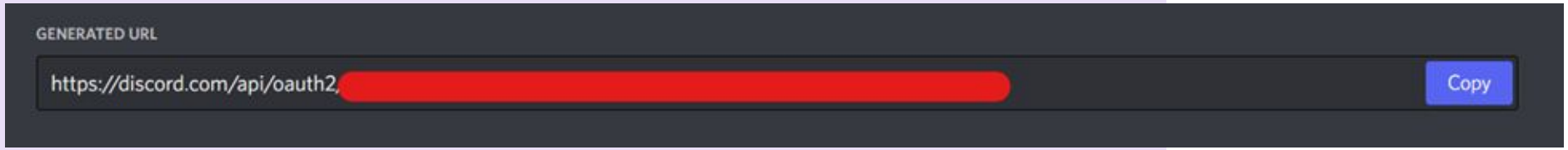
# Send your Bot to your Server

In OAuth –> URL Generator



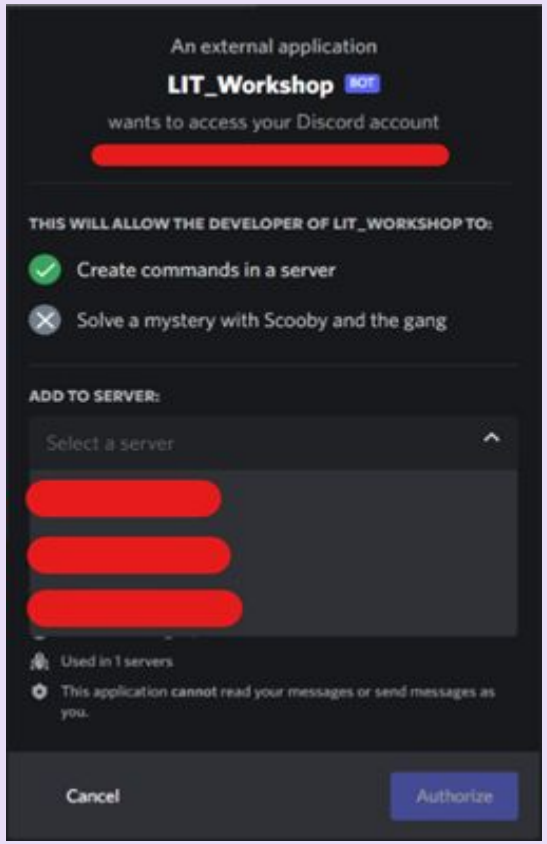
# Send your Bot to your Server

After setting permissions, copy generated URL into your browser.



# Send your Bot to your Server

Select the server you want to send your bot into and authorize it.



# Coding hands-on

# Importing Libraries

```
import os
from dotenv import load_dotenv
import requests
import json
import discord
from discord.ext import commands
```

- Those are the libraries needed for today's workshop, which has been installed earlier on.
- discord for all of discord bot functions and dotenv to load environmental variables in .env file.
- The rest of the libraries are for what our discord bot does.

# Initialising our Bot and Environmental Variables

```
load_dotenv()

# registering a command with $ prefix
client = commands.Bot(command_prefix="$",
intents=discord.Intents.all())

token = os.getenv('DISCORD_TOKEN')
```

- load_dotenv() loads variables in .env as environment variables.
- commands.Bot represents the discord bot.
- intents=discord.Intents.all() allows bot to access all permissions given.
- command_prefix is what the message must contain to have a command invoked.
- getenv() accesses token from environment variables.

# Activating and Ensuring our Bot is Live

```
@client.event
async def on_ready():
    print("Logged in as a bot
{0.user}".format(client))
```

```
# activating the bot
client.run(token)
```

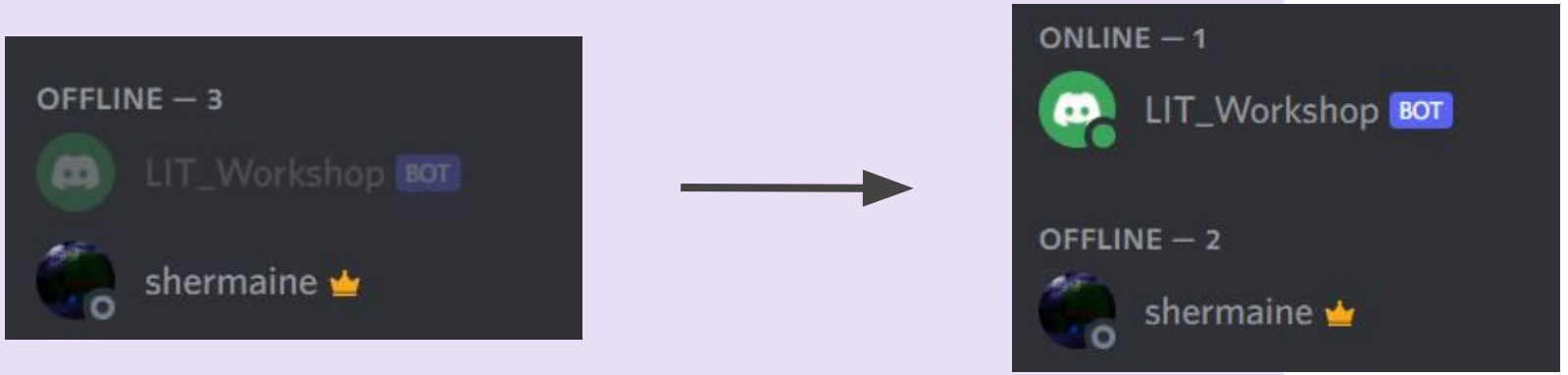
- @client.event is a decorator from the discord library for event handlers.
- “async def on_ready()” defines on_ready asynchronous function from discord library. This function is default from discord and should not be changed.
- To ensure our bot is working, this final statement will activate the bot to run, based on whatever we have coded.

# Ensuring Our Bot's Connection

```
2022-09-21 22:07:13 INFO  discord.client logging
in using static token
2022-09-21 22:07:14 INFO  discord.gateway Shard
ID None has connected to Gateway (Session ID:
dc66488d2c22d67ec4736d7a7aee7396).
Logged in as a bot LIT_Workshop#3202
```

- When activating the bot, this should be the expected output, as a result of the on_ready function.

# Ensuring Our Bot is Live (on Discord)



# Break

# Kahoot Time

# Let’s test your understanding so far!

Join at: https://www.menti.com/alka58wjsw81

Game Pin:

**8386 1114**

QR Code:

# What Functions Will We Code Today?

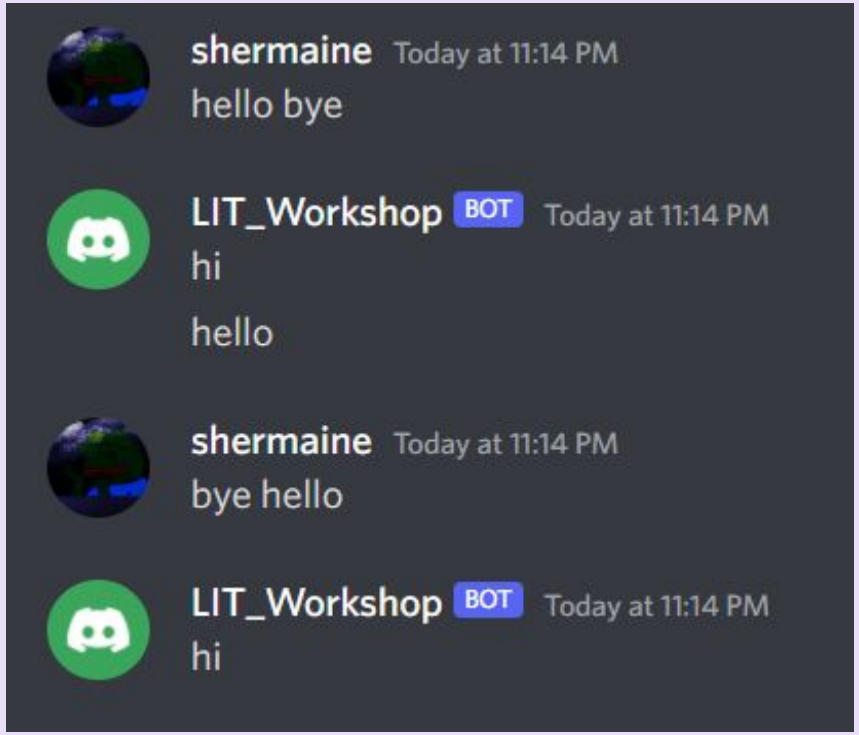
1. **On_message Function**
   a. If “hello” exists in message.
   b. If message startswith “hello”.
2. **Discord Commands**
3. **Integrating Lists and API**
   a. Initialise list of words to trigger the bot.
   b. Get quotes from an API.
   c. Activating API function.

# Getting the Bot to Respond to Messages

```
@client.event
async def on_message(message):
    if message.author == client.user:
        return
    # get the message content & username
    username = str(message.author).split("#")[0]
    user_message = message.content

    # if hello in message, reply with hi
    if 'hello' in user.message:
        await message.channel.send('hi')
   # if message startswith hello, reply hello
    if user_message.startswith("hello"):
        await message.channel.send('hello')
```

- message.author == client.user checks if the bot is the author.
- 'hello' in user.message checks if hello exists in the user's message.
- user_message.startswith("hello") checks if the message starts with hello.
- await message.channel.send() makes the bot send passed message into the channel.

# Testing the bot




- Here, we test the bot in the discord server that it was added to.
- Both the messages we sent had “hello” so the bot replies “hi” for both.
- However, the second message does not start with hello so the startswith function will not be activated, only activating the “hello in user_message” command.

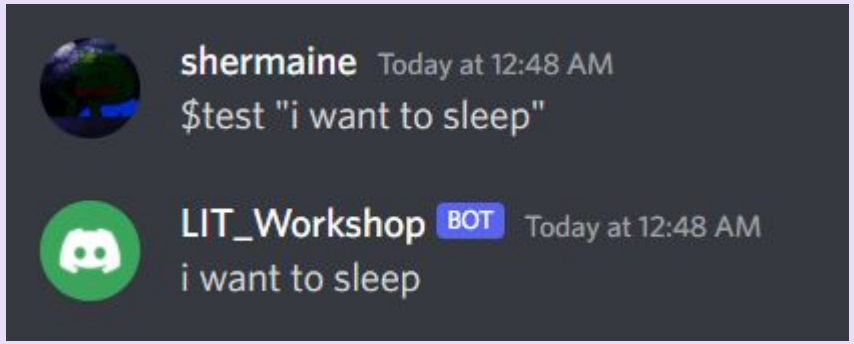
# Getting the Bot to Respond to Commands

```
@client.event
async def on_message(message):
    if user_message.startswith("hello"):
        await message.channel.send('hello')

await client.process_commands(message)
```

```
# returning arguments if command detected
@client.command()
async def test(ctx, arg):
    await ctx.send(arg)
```

- Return to the on_message function and adds a line after the hello response.
- client.command overrides the original function, hence, the await statement avoids override.
- Create a new test function for commands, where ctx = client.channel
- Commands repeat the message sent.

# Testing the Bot




- In this section, we test the command function by starting the message with $test, which is what we have set as our command function.
- The sentence sent AFTER the activated command will then be repeated by the bot.
- Note: if “” are not used, it will only send the first word. Eg: “i”

# Bonus!
## Integrating API and Lists

# Initialise a List of Words

```
# setting list of sad words
sad_words = ["sad", "depressed", "unhappy",
"angry", "miserable"]
```

- Here, we initialise a list of trigger words for the bot to reply to.
- Whenever the user uses any of the specific words in their channel, the bot will be triggered to send a reply.
- We will cover more in a later segment.

# Load inspirational quotes from API

```
def get_quote():
   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
```

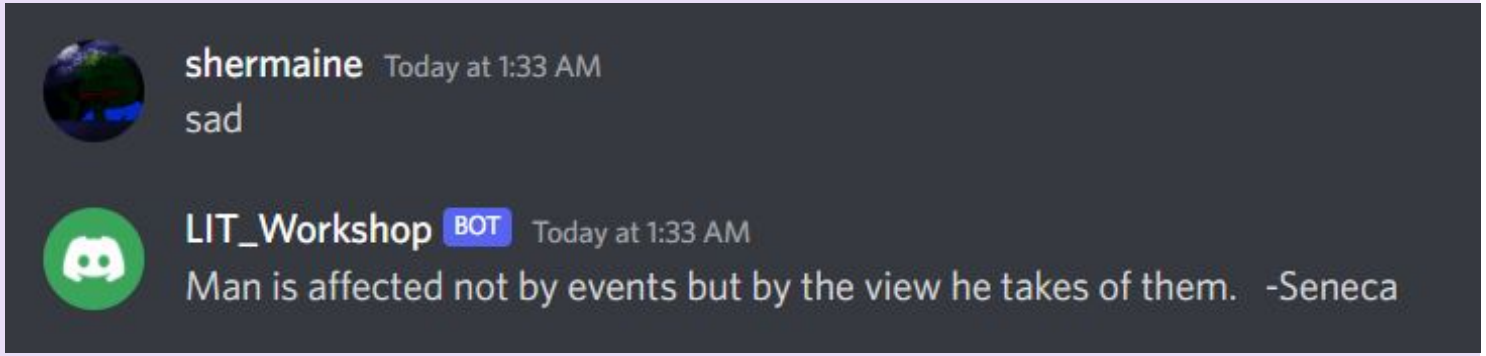
- This function gets and returns randomised quotes from an API.
- requests.get() sends and returns the results of a http get request.
- json.loads() parses the json format results.
- The data is string formatted into quotes - author then returned.

*API Link: https://zenquotes.io/api/random*

# Activating API using Lists Predefined Lists

```
if any(word in user_message for word in
sad_words):
   await message.channel.send(get_quote())
```

- The is a conditional statement, which checks if any of the trigger sad words from the list are in the users' message.
- If True, the bot will activate the quote function we created earlier.



# Deployment

# Create Files for Deployment

requirements.txt

```
discord==2.0.0
discord.py==2.0.1
python-dotenv==0.20.0
requests==2.28.1
```

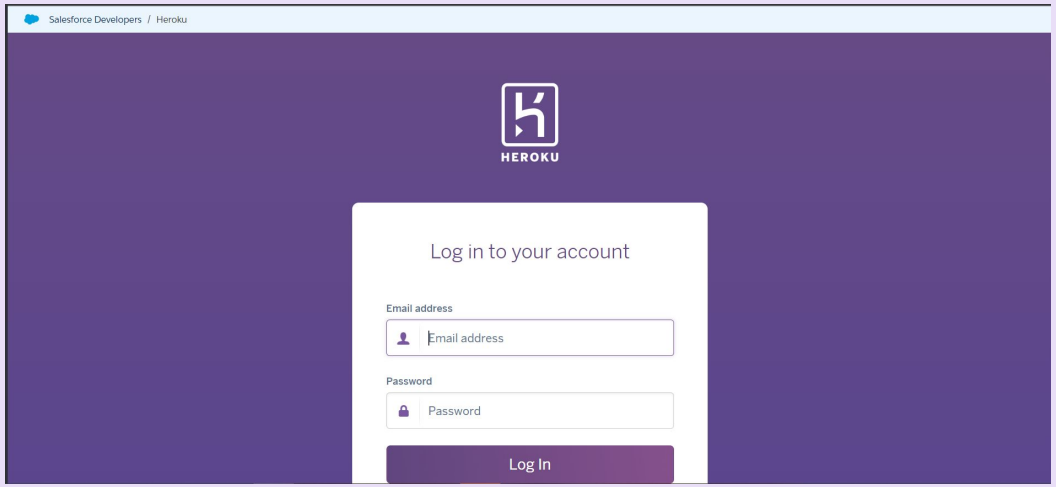
Procfile

```
worker: python main.py
```

1. Create requirements.txt file
   a. pip freeze > requirements.txt

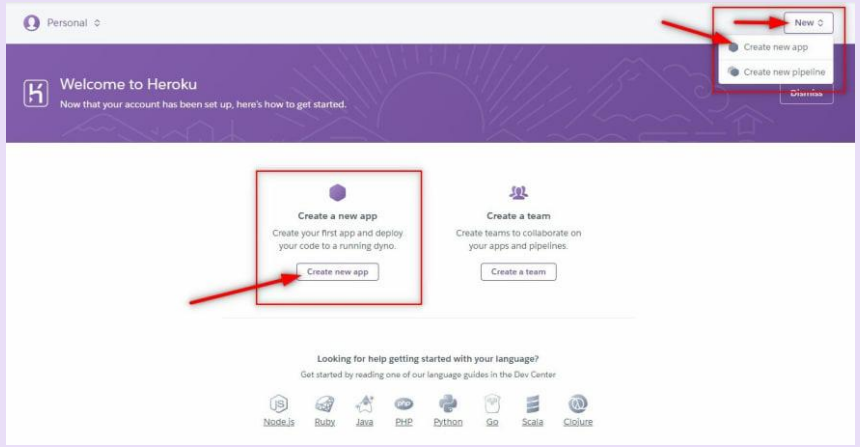
2. Create Procfile (Note: has to be this exact naming)
   a. In Procfile, add “worker: python main.py”

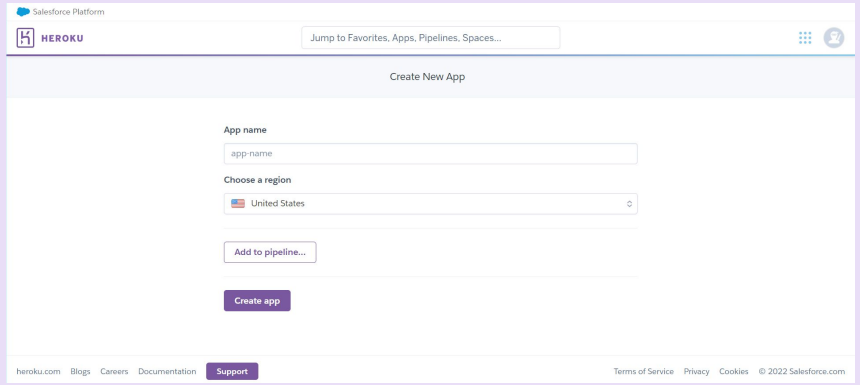
# Heroku account




- *Heroku link: https://www.heroku.com/*
- Login to your heroku account.

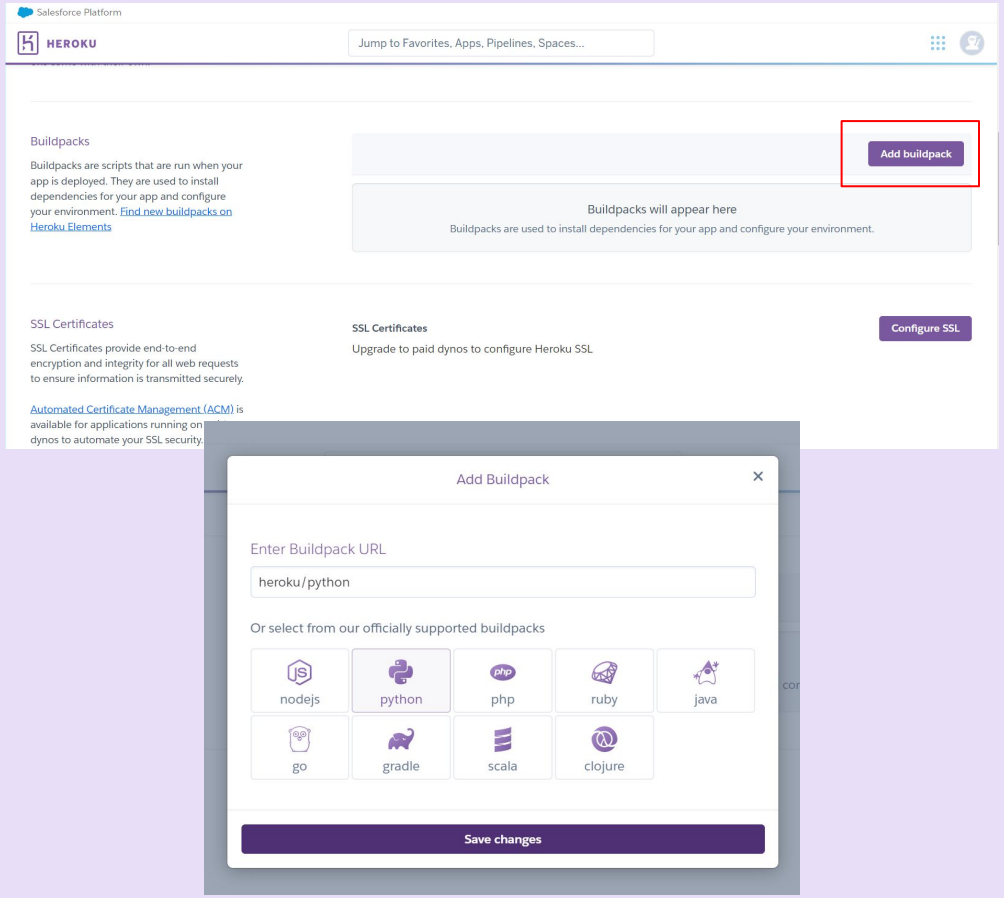
# Create new app



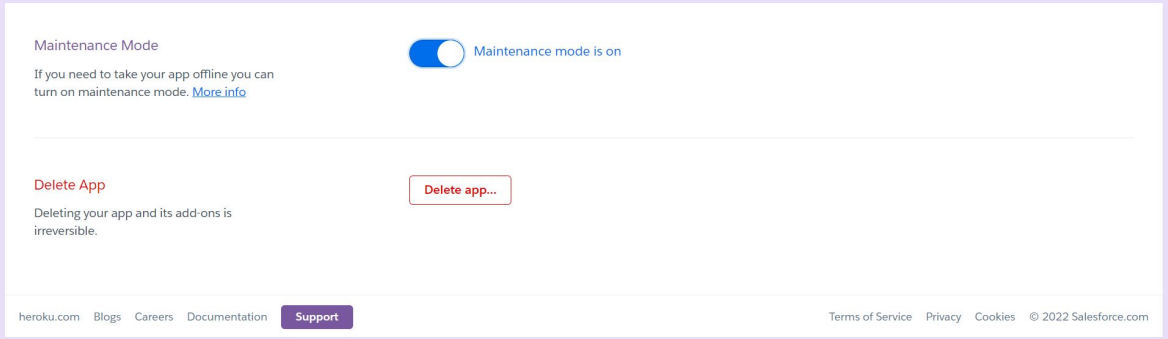

- Create a new heroku app by clicking on create new app or new > create new app.
- Give your app any name you want.
- Choose United States as your region.

# Add a buildpack



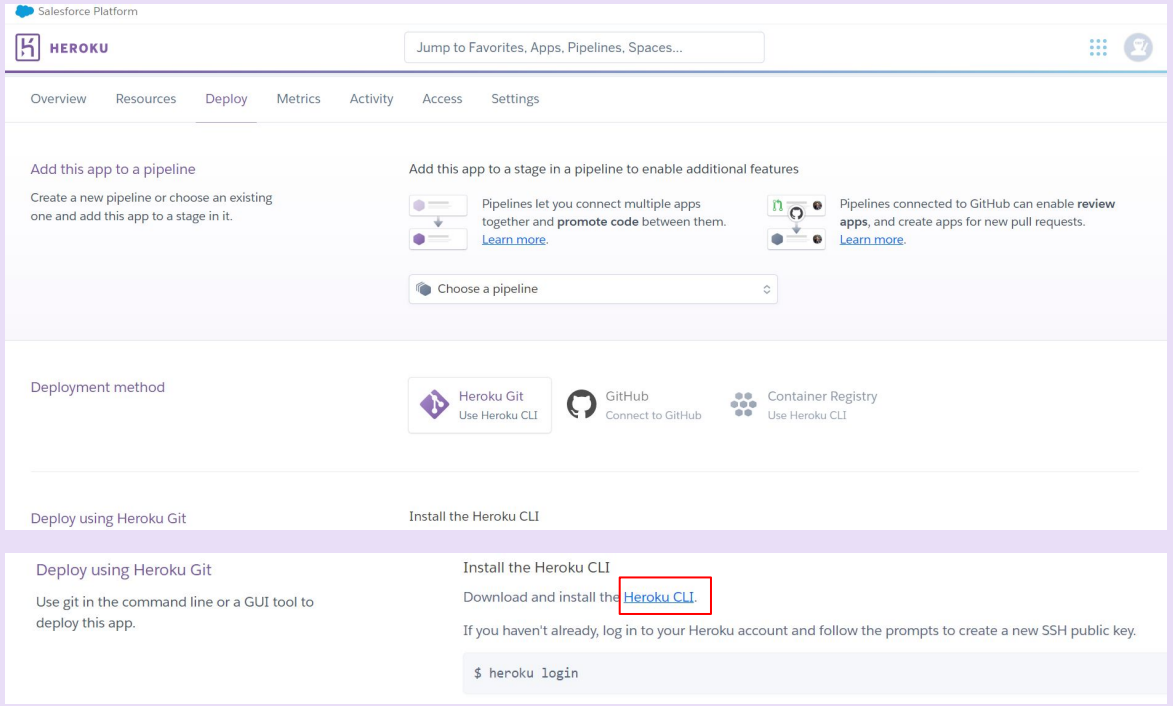

- Go to settings.
- Scroll down to the buildpacks section.
- Click add buildpack.
- In the popup, select python buildpack and save changes.

# Turn on maintenance mode




- In settings, scroll down to maintenance mode
- Click on the toggle to turn on maintenance mode

# Installing Heroku CLI




- Go to deploy.
- Choose Heroku Git as deployment method.
- Click on the Heroku CLI link.
- Install Git and heroku CLI according to your os. (Mac/ Win)

# Login and Create New Repository

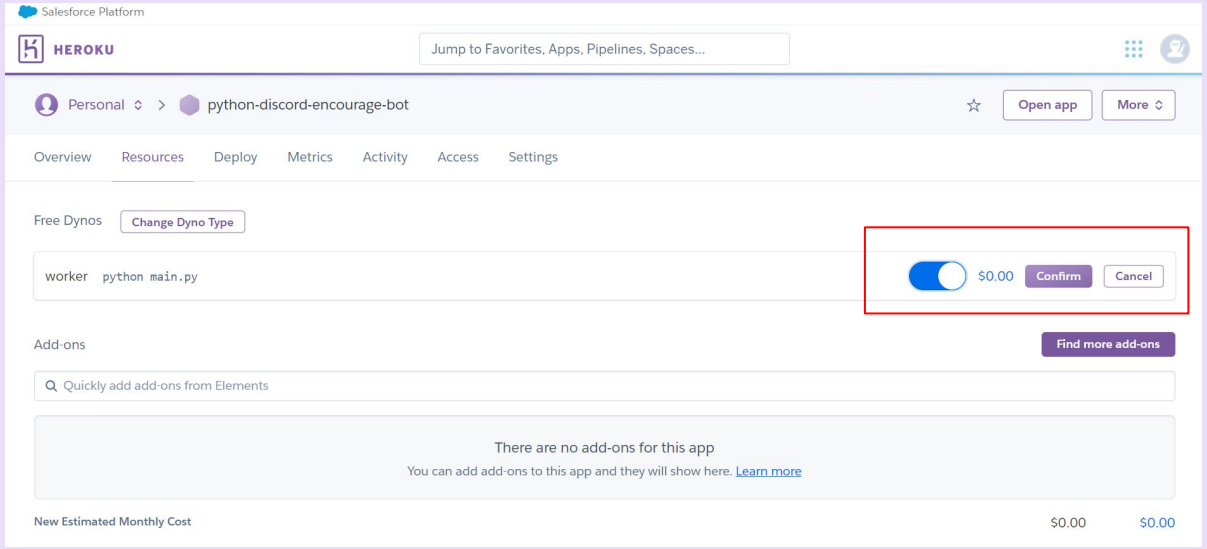
```
PS C:\Year 2\NYPLIT> heroku login -i
heroku: Enter your login credentials
Email [pehshermaine@gmail.com]: pehshermaine@gmail.com
Password: **********
Logged in as pehshermaine@gmail.com
```

```
PS C:\Year 2\NYPLIT> git init
PS C:\Year 2\NYPLIT> heroku git:remote -a litdeploy
```

- Login to heroku on your browser or terminal using
  - heroku login (browser)
  - heroku login –i (terminal)
- Ensure you are in your project folder path.
- Initialize a git repository using “git init”.
- Add heroku git.
  - heroku git:remote –a heroku–app–name

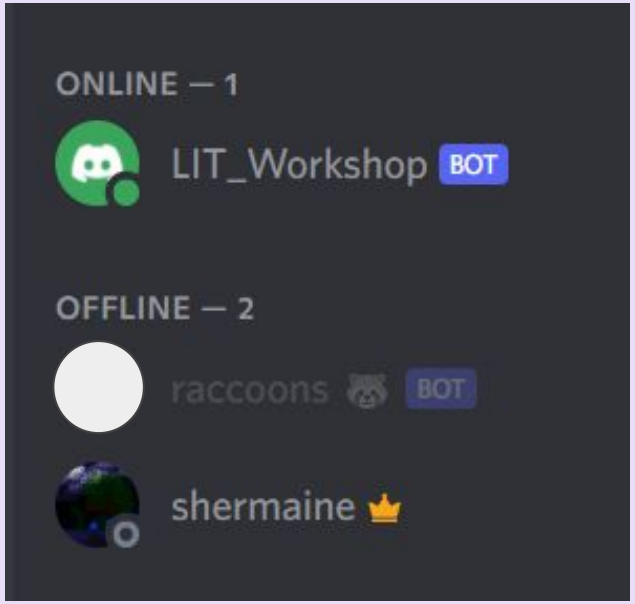
# Deploy discord bot heroku app

```
PS C:\Year 2\NYPLIT> git add .
PS C:\Year 2\NYPLIT> git commit -m “deploy”
PS C:\Year 2\NYPLIT> git push heroku master
```




- Commit deploy to heroku using Git commands
  1. git add .
  2. git commit -am “make it better”
  3. git push heroku master
- After deploy, go to resources.
- Enable the free dynos by editing the toggle and confirm.

# Discord bot online




- Previously, our bot goes live only when running the code.
- However, upon successful deployment, we can see that our bot is now live on discord, even without running the code.
- Deployment allows our bot to stay live at all times, responding to messages, even if you are not controlling it at the moment.

# Thank you!

**Do you have any questions?**

Hop over to the get-support voice channel! We will be there till the end of the event, or feel free to drop us a message in get-support text channel

https://nyp-lit.github.io/

@nyp_lit