```python
import discord
import os
import yaml
import random
from typing import Optional, Literal
from discord.ext import commands
from discord.ext.commands import Context
from discord.ext.commands import Greedy
from discord import app_commands

intents = discord.Intents.all()
bot = commands.Bot(command_prefix="/", intents=intents)  # Replace '!' with your command prefix

@bot.event
async def on_command_error(ctx, error):
    if isinstance(error, commands.CommandError):
        await ctx.send('Sorry, I did not understand that command. Please try again.' )

@bot.event
async def on_ready():
    print(f"We have logged in as {bot.user}")
    print("Registered commands:")
    for command in bot.commands:
        print(command.name)
    for command in city.commands:
        print(command.name)
    for command in media.commands:
        print(command.name)
    for command in sense.commands:
        print(command.name)
    for command in npc.commands:
        print(command.name)
    for command in matrix.commands:
        print(command.name)
    for command in runner.commands:
        print(command.name)
    await bot.wait_until_ready()

#==========================================
#                   Sync commands
#==========================================

@bot.command()
async def sync(
  ctx: Context, guilds: Greedy[discord.Object], spec: Optional[Literal["~", "*", "^"]] = None) -> None:
    if not guilds:
        if spec == "~":
            synced = await ctx.bot.tree.sync(guild=ctx.guild)
        elif spec == "*":
            ctx.bot.tree.copy_global_to(guild=ctx.guild)
            synced = await ctx.bot.tree.sync(guild=ctx.guild)
        elif spec == "^":
            ctx.bot.tree.clear_commands(guild=ctx.guild)
            await ctx.bot.tree.sync(guild=ctx.guild)
            synced = []
        else:
            synced = await ctx.bot.tree.sync()

        await ctx.send(
            f"Synced {len(synced)} commands {'globally' if spec is None else 'to the current guild.'}"
        )
        return

    ret = 0
    for guild in guilds:
        try:
```

```
            await ctx.bot.tree.sync(guild=guild)
        except discord.HTTPException:
            pass
        else:
            ret += 1

    await ctx.send(f"Synced the tree to {ret}/{len(guilds)}.")
    print(f"Synced the tree to {ret}/{len(guilds)}.")

#commands.Greedy`
#discord.Object`
#typing.Optional` and `typing.Literal`

#Works like:
#!sync` -> global sync
#!sync ~` -> sync current guild
#!sync *` -> copies all global app commands to current guild and syncs
#!sync ^` -> clears all commands from the current guild target and syncs (removes guild commands)
#!sync id_1 id_2` -> syncs guilds with id 1 and 2

#=======================================
#              Setup
#=======================================

# it is used for the cooldown to prevent the bot from spam attack
@bot.event
async def on_command_error(ctx, error):
    if isinstance(error, commands.CommandOnCooldown):
        await ctx.send(f"**Try after {round(error.retry_after, 2)} seconds.")

# This is your roll logic function
async def roll_logic(ctx, tables, message_format, num_rolls=1):
    with open("RandomRolls.yaml", 'r') as stream:
        try:
            roll_tables = yaml.safe_load(stream)
            if roll_tables:
                results = []
                for table in tables:
                    table_results = []
                    for _ in range(num_rolls):
                        result = random.choice(list(roll_tables[table].values()))
                        table_results.append(result)
                    results.append(" and ".join(table_results))

                message = message_format.format(*results)
                await ctx.send(message)
        except yaml.YAMLError as exc:
            print(exc)

@bot.hybrid_group(name='city', help="Commands about urban life, buildings, streets and cars.")
async def city(ctx):
    await ctx.send('Invalid sub command passed...')

@bot.hybrid_group(name='media', help="Commands about generating TV, Music, AR and Ads related content.")
async def media(ctx):
    await ctx.send('Invalid sub command passed...')

@bot.hybrid_group(name='npc', help="Commands about generating NPC's.")
async def npc(ctx):
    await ctx.send('Invalid sub command passed...')

@bot.hybrid_group(name='matrix', help="Commands about generating Matrix and Data related content.")
async def matrix(ctx):
    await ctx.send('Invalid sub command passed...')

@bot.hybrid_group(name='sense', help="Commands about generating sensory experiences like
```

```
    smell, sight, sounds etc.")
133 async def sense(ctx):
134     await ctx.send('Invalid sub command passed...')
135 
136 @bot.hybrid_group(name='runner', help="Commands about generating missions, runner life things
    and other content.")
137 async def runner(ctx):
138     await ctx.send('Invalid sub command passed...')
139 
140 # ================================================================
141 #                         Roll Commands
142 # ================================================================
143 
144 @city.command(name='buildings', help="Description of a random building with type, features,
    style, its current state, quirky feature and secret" )
145 async def buildings(ctx):
146     tables = [
147         "1_Building_Type",
148         "1_Building_Feature",
149         "1_Quirky_Style",
150         "1_Quirky_State",
151         "1_Quirky_Feature",
152         "1_Quirky_Secret",
153     ]
154     message_format = "You see a {} that is {} with a {} style and {} look. There are {} and a
    secret {}."
155     await roll_logic(ctx, tables, message_format)
156 
157 
158 @sense.command(name='smell', help="Desribing a random smell. Add a number after the command
    to generate multiple smells")
159 async def smell(ctx, num_rolls: int = 1):
160     tables = ["8_Smell"]
161     message_format = "You smell {}."
162     await roll_logic(ctx, tables, message_format, num_rolls)
163 
164 
165 @npc.command(name='conflictgroup', help="Generate a random group and what kind of conflict
    they have with another group")
166 async def conflictgroup(ctx):
167     tables = [
168         "2_Conflict_Group",
169         "2_Conflict_Source",
170         "2_Conflict_Opposing",
171     ]
172     message_format = "The group {} are/is in conflict because {} with/against {}."
173     await roll_logic(ctx, tables, message_format)
174 
175 
176 @city.command(name='business', help="Generate a random small street business. It describes
    the quality, the current status, the type and what kind of security it has" )
177 async def business(ctx):
178     tables = [
179         "3_AltBusiness_Quality",
180         "3_AltBusiness_Status",
181         "3_AltBusiness_Type",
182         "3_AltBusiness_Security",
183     ]
184     message_format = "A {} and {} {}. There is/are {} for security on the premises."
185     await roll_logic(ctx, tables, message_format)
186 
187 
188 @city.command(name='streetfinds', help="Generates a random odd or weird 'thing' you can find
    on the streets")
189 async def streetfinds(ctx):
190     tables = ["4_Weird_Street_Finds"]
191     message_format = "You stumble upon a {}."
192     await roll_logic(ctx, tables, message_format)
193 
```

```python

@city.command(name='legacyinfrastructure', help="Rolls a random legacy building or part of old infrastructure and what is special about it")
async def legacyinfrastructure(ctx):
    tables = ["5_Legacy_Infrastructure"]
    message_format = "You stumble upon a {}."
    await roll_logic(ctx, tables, message_format)


@matrix.command(name='legacydata', help="Rolls a random type of legacy Data device or paydata and what is special about it")
async def legacydata(ctx):
    tables = ["6_Legacy_Data"]
    message_format = "You stumble upon a {}."
    await roll_logic(ctx, tables, message_format)


@city.command(name='nightlife', help="Rolls a random nightlife location, it's status, the type of security it has and what the vibe of the place is")
async def nightlife(ctx):
    tables = [
        "7_Nightlife_Location",
        "7_Nightlife_Status",
        "7_Nightlife_Security",
        "7_Nightlife_Vibe",
    ]
    message_format = "The nightclub '{}' has/is {}. The on-site security is/are {}. The vibe of the place is {}."
    await roll_logic(ctx, tables, message_format)

@media.command(name='music', help="Rolls a random futuristic music genre")
async def music(ctx):
    tables = ["25_Music_Genre"]
    message_format = "You would describe the music genre as {}"
    await roll_logic(ctx, tables, message_format)

@sense.command(name='sound', help="Describing a random sound. Add a number after the command to roll multiple sounds")
async def sound(ctx, num_rolls: int = 1):
    tables = ["8_Sounds"]
    message_format = "You hear {}"
    await roll_logic(ctx, tables, message_format, num_rolls)

@sense.command(name='gutfeeling', help="Describing a random gutfeeling. Add a number after the command to roll multiple gutfeelings")
async def gutfeeling(ctx, num_rolls: int = 1):
    tables = ["8_Gut_Feeling"]
    message_format = "You can't shake the feeling that {}"
    await roll_logic(ctx, tables, message_format, num_rolls)

@sense.command(name='sight', help="Describing a random sight. Add a number after the command to roll multiple sights")
async def sight(ctx, num_rolls: int = 1):
    tables = ["8_Sights"]
    message_format = "You see {}"
    await roll_logic(ctx, tables, message_format, num_rolls)

@media.command(name='ad', help="Roll a random advertisement with its marketing style, the brand, product and product line")
async def ad(ctx):
    tables = [
        "9_Infotainment_Marketing_Style",
        "9_Infotainment_Brand",
        "9_Infotainment_Range",
        "9_Infotainment_Product_Line",
    ]
    message_format = "You see a {} commercial for {} {} {} product."
    await roll_logic(ctx, tables, message_format)

```

```
@city.command(name='cars', help="Roll a random car on the street")
async def cars(ctx, num_rolls: int = 1):
    tables = ["10_Road_Vehicles"]
    message_format = "You see {}"
    await roll_logic(ctx, tables, message_format, num_rolls)

@media.command(name='ar', help="Roll a random augmented reality AR-Icon with its type, the aesthetic and image style")
async def ar(ctx):
    tables = [
        "11_AR_Type",
        "11_AR_Aesthetic",
        "11_AR_Image_Style",
    ]
    message_format = "The AR image is about {} with {} and {}."
    await roll_logic(ctx, tables, message_format)

@npc.command(name='citizen', help="Rolls a random citizen with its first impression, how they looks, what style they have, the vibe they give and what kind of accessories they have" )
async def citizen(ctx):
    tables = [
        "12_Instacitizen",
        "12_Instacitizen_Impression",
        "12_Instacitizen_Looks",
        "12_Instacitizen_Style",
        "12_Instacitizen_Vibe",
        "12_Instacitizen_Accessories",
    ]
    message_format = "He/She is a {}. Your first impression is {} with {}. He/She has {} and their vibe is {}. What's special about them is, that they are/have {}"
    await roll_logic(ctx, tables, message_format)

@npc.command(name='needs', help="Rolls a random nefarious need of an NPC and how badly they need to have it / what they're willing to do for it" )
async def needs(ctx):
    tables = [
        "13_Wants",
        "13_Level_Of_Need",
    ]
    message_format = "**Wants**: {}. **Level of need**: {}"
    await roll_logic(ctx, tables, message_format)

@npc.command(name='tattoo', help="Roll a random tattoo with its style, what motif it has and where it is tattooed on the body")
async def tattoo(ctx):
    tables = [
        "14_Tattoo_Style",
        "14_Tattoo_What",
        "14_Tattoo_Where"
    ]
    message_format = "The tattoo {} {}. It is tattooed {}."
    await roll_logic(ctx, tables, message_format)

@npc.command(name='streetwalker', help="Roll a random streetwalker, what genderidentity they have and a short description")
async def streetwalker(ctx):
    tables = [
        "15_Streetwalker_Genderidentity",
        "15_Streetwalker_Type",
    ]
    message_format = "You see a {} streetwalker, {}"
    await roll_logic(ctx, tables, message_format)

@npc.command(name='salaryman', help="Roll a random corporate salaryman with its name, their job title, their quirk and how they look" )
async def salaryman(ctx):
    tables = [
        "16_Corpo_Name",
        "16_Corpo_Surname",
```

```
317             "16_Corpo_Job",
318             "16_Corpo_Quirk",
319             "16_Corpo_Look",
320         ]
321         message_format = "They introduce themselves as {} {}. They work as {} and their quirk is
            {}. They can be described as {}"
322         await roll_logic(ctx, tables, message_format)
323
324     @runner.command(name='corpomission', help="Roll a random corporate mission, with what needs
        to be done, what additional ressources the corporation can give the runners and what the
        unexpected twist would be")
325     async def corpomission(ctx):
326         tables = [
327             "17_Corpo_Mission",
328             "17_Corpo_Ressources",
329             "17_Corpo_Twist",
330         ]
331         message_format = "**Job**: \n- {}. \n- {}. \n- {}"
332         await roll_logic(ctx, tables, message_format)
333
334     @runner.command(name='randomevent', help="Roll a random event that can happy any time and
        anywhere")
335     async def randomevent(ctx, num_rolls: int = 1):
336         tables = ["17_Random_Event"]
337         message_format = "**Random Event**: {}"
338         await roll_logic(ctx, tables, message_format, num_rolls)
339
340     @runner.command(name='trap', help="Rolls a random trap for the players to run into. Add
        numbers after the command to roll multiple traps" )
341     async def trap(ctx, num_rolls: int = 1):
342         tables = ["17_BoobyTrap"]
343         message_format = "**Booby Trap**: {}"
344         await roll_logic(ctx, tables, message_format, num_rolls)
345
346     @npc.command(name='gang', help="Roll a random gang with their name, what their business is,
        what else they deal in and what is rumored about them on the street" )
347     async def gang(ctx):
348         tables = [
349             "18_Gang_Name1",
350             "18_Gang_Name2",
351             "18_Gang_Name3",
352             "18_Gang_Name4",
353             "18_Gang_Activity",
354             "18_Gang_Deal",
355             "18_Gang_Rumor"
356         ]
357         message_format = "**They are known as {} {} {} {}**. \nTheir business is: {}, and their
            deal is: {}. It is rumored that: {}."
358         await roll_logic(ctx, tables, message_format)
359
360     @npc.command(name='police', help="Roll a random law enforcement unit, with what type they
        are, what they're currently doing and how they handle the situation" )
361     async def police(ctx):
362         tables = [
363             "19_Police_Department",
364             "19_Police_Type",
365             "19_Police_Jobs",
366             "19_Police_Response_Level",
367         ]
368         message_format = "{} has sent {}. They are {}. {}."
369         await roll_logic(ctx, tables, message_format)
370
371     @npc.command(name='policebackup', help="Roll random law enforcement backup, with what type
        they are and which tactics they are going to use to solve the situation" )
372     async def policebackup(ctx):
373         tables = [
374             "19_Police_Backup",
375             "19_Police_Tactics",
376         ]
```

```
377         message_format = "The law enforcers on site have requested backup. The HQ sends {} and they're going to use {}"
378         await roll_logic(ctx, tables, message_format)
379
380     @npc.command(name='fixer', help="Roll a random fixer contact, with their type, their actual job, their circumstances and how they look")
381     async def fixer(ctx):
382         tables = [
383             "20_Fixer_Type",
384             "20_Fixer_Job",
385             "20_Fixer_Circumstances",
386             "20_Fixer_Job",
387             "20_Fixer_Look"
388         ]
389         message_format = "Your Fixers name is {} and their job is being a {}. They/They're {} {}. Your first impression of them is: {}"
390         await roll_logic(ctx, tables, message_format)
391
392     @npc.command(name='hiredgun', help="Roll a random hired gun, with what weapon they carry, their clothing, their circumstances and type")
393     async def hiredgun(ctx):
394         tables = [
395             "24_Hired_Gun",
396             "24_Hired_Gun_Weapon",
397             "24_Hired_Gun_Clothing",
398             "24_Hired_Gun_Circumstance",
399             "21_Client_Type"
400         ]
401         message_format = "The hired gun introduces themselves as {}. They're wielding as a weapon a {}. Your first impression of them is: {}. And in the past: {} {}."
402         await roll_logic(ctx, tables, message_format)
403
404     @runner.command(name='mrjohnsonjob', help="Roll a random job from a Mr.Johnson, with what type of Mr.Johnson they are, what they want, what the target is and the hidden twist that awaits the players")
405     async def mrjohnsonjob(ctx):
406         tables = [
407             "21_MrJohnson_Type",
408             "21_MrJohnson_Want",
409             "21_MrJohnson_Target",
410             "21_MrJohnson_Action",
411             "17_Corpo_Twist",
412         ]
413         message_format = "The Johnsons introduces himself as a {}. He/She {} a {}. To achieve this goal he needs the players to {} someone. {}"
414         await roll_logic(ctx, tables, message_format)
415
416     @matrix.command(name='datadevice', help="Roll a random Data Device, with what kind of content is on it and what the history of it was")
417     async def datadevice(ctx):
418         tables = [
419             "22_DataDevice_Content",
420             "22_DataDevice_History",
421         ]
422         message_format = "On the Data Device you find {}. Given the data on it, it seems to be {}"
423         await roll_logic(ctx, tables, message_format)
424
425     @matrix.command(name='sota', help="Roll a random State-of-the-Art Device and what condition it is in")
426     async def sota(ctx):
427         tables = [
428             "22_StateOfTheArt_Type",
429             "22_StateOfTheArt_Condition",
430         ]
431         message_format = "You find a {}. Its condition/specifications is {}."
432         await roll_logic(ctx, tables, message_format)
433
434     @runner.command(name='clutter', help="Roll a random piece of useless clutter/items. Add a number after the command to roll multiple clutter items")
```

```python
async def clutter(ctx, num_rolls: int = 1):
    tables = ["23_Corpse_Object"]
    message_format = "You find: {}"
    await roll_logic(ctx, tables, message_format, num_rolls)

@runner.command(name='corpse', help="Roll a random description for a corpse and what condition it is in or what could have been the circumstances of the death" )
async def corpse(ctx):
    tables = [
        "23_Corpse_Description",
    ]
    message_format = "You look at the corpse in front of you. You find:  '*{}*'"
    await roll_logic(ctx, tables, message_format)

@npc.command(name='client', help="Roll a random type of client, with what type they are, what they want etc.")
async def client(ctx):
    tables = [
        "21_Client_Type",
        "21_Client_Want",
        "21_Client_Action",
        "21_Client_Item",
    ]
    message_format = "The client is a {} and and they {} {} a {}."
    await roll_logic(ctx, tables, message_format)

@media.command(name='socialmedia', help="Roll random social media content with description about what it is about. Add a number after the command to roll multiple social media contents" )
async def socialmedia(ctx, num_rolls: int = 1):
    tables = ["25_Social_Media"]
    message_format = "You check out the social media on the matrix. You summarise the content as: {}."
    await roll_logic(ctx, tables, message_format, num_rolls)

@media.command(name='tv', help="Roll a random TV-Show. Add a number after the command to roll multiple shows")
async def tv(ctx, num_rolls: int = 1):
    tables = ["25_TV_Shows"]
    message_format = "You stare at the screen, watching a TV-show. The show is called {}."
    await roll_logic(ctx, tables, message_format, num_rolls)

@city.command(name='atypicalweather', help="Roll a random type of dangerous or atypical weather to annoy your players")
async def atypicalweather(ctx):
    tables = [
        "25_Atypical_Weather",
    ]
    message_format = "You check your surroundings. The Weather is unusual today. You would describe it as: {}."
    await roll_logic(ctx, tables, message_format)

@runner.command(name='insult', help="Just like in Monkey-Island. Throw random insults at your players. Add a number after the command to insult them multiple times" )
async def insult(ctx, num_rolls: int = 1):
    tables = ["26_Insults"]
    message_format = "You seem to have offended the Person in front of you. They tell you: '*{}*'"
    await roll_logic(ctx, tables, message_format, num_rolls)

@city.command(name='street', help="Roll a random street with detailed description about what condition it is in, what kind of AR content is there and how the buildings look like" )
async def street(ctx):
    tables = [
        "27_Street_Description",
        "27_Street_Condition",
        "27_Augmented_Reality_Ads",
        "27_Building_Sights",
    ]
```

```
493         message_format = "You see {} {} {} {}"
494         await roll_logic(ctx, tables, message_format)
495 
496     @city.command(name='publictransport', help="Roll a random public transport vehicle, what condition it is in, the type of passengers on it, what it sounds/smells like and what the driver looks like")
497     async def publictransport(ctx):
498         tables = [
499             "27_Public_Transport_Vehicles",
500             "27_Public_Transport_Condition",
501             "27_Public_Transport_Passengers",
502             "27_Public_Transport_Passengers",
503             "27_Public_Transport_Passengers",
504             "27_Public_Transport_Sensory",
505             "27_Public_Transport_Driver",
506         ]
507         message_format = "Your public transport vehicle is a {}. {} Among the passengers you see {} {} {}.{} It is driven by {}"
508         await roll_logic(ctx, tables, message_format)
509 
510     @runner.command(name='publicencounter', help="Roll a random public encounter on the street. Just be aware, most of them include addicts, crazy people or other nuisances for your players")
511     async def publicencounter(ctx):
512         tables = [
513             "27_Public_Encounter",
514             "27_Public_Encounter_Scam",
515         ]
516         message_format = "{} {}"
517         await roll_logic(ctx, tables, message_format)
518 
519     #=========================================
520     #               END
521     # ========================================
522 
523     # Token
524     token = os.getenv('TOKEN')
525     bot.run(token)
526 
```