# ECSC Estonia Prequalifier - DDC Admin Bot

This was my favourite challenge of the `ECSC Estonia Prequalifier`.

We are given the discord bot's source code:

```
#!/bin/python3
import discord
from discord.ext import commands
import os

# Intents setup
intents = discord.Intents.default()
intents.message_content = True
intents.members = True  # Required to access member updates and role
management

# Bot prefix and intents
bot = commands.Bot(command_prefix="!", intents=intents)

# IDs for roles and channel names
TOKEN = os.getenv('DISCORD_TOKEN')
MODERATOR_ROLE_NAME = "moderator"
MEMBER_ROLE_NAME = "member"
VERIFICATION_CHANNEL_NAME = "verification"


@bot.event
async def on_ready():
    print(f'Bot {bot.user.name} has connected to Discord!')

async def get_or_create_verification_channel(guild):
    # Look for the verification channel by name
    verification_channel = discord.utils.get(guild.text_channels,
name=VERIFICATION_CHANNEL_NAME)

    # If the channel doesn't exist, create it
    if verification_channel is None:
        print(f"Verification channel not found. Creating a new one...")
        verification_channel = await guild.create_text_channel(
            VERIFICATION_CHANNEL_NAME,
            topic="Channel for moderator to verify users",
            reason="Create verification channel for user verification",
        )
        print(f"Created verification channel:
{verification_channel.name}")
```

```
    return verification_channel

# Command to request verification
@bot.command(name='verifyme')
async def request_verification(ctx):
    user = ctx.author
    guild = ctx.guild

    # Get the moderator and member roles
    mod_role = discord.utils.get(guild.roles, name=MODERATOR_ROLE_NAME)
    if mod_role is None:
        await ctx.send("Moderator role not found in the server.")
        return

    # Get or create the verification channel
    verification_channel = await get_or_create_verification_channel(guild)

    # Notify the user their request has been sent
    await ctx.send(f"{user.mention}, your verification request has been
sent to the moderators.")
    # Send the verification request to the verification channel
    msg = f"{user.mention} has requested verification. Moderators with
{mod_role.mention}, please verify."

    verification_msg = await verification_channel.send(msg)

    # Ask moderators to react to verify the user
    await verification_msg.add_reaction("✅")

    # Create a check function to confirm it's the correct reaction from a
moderator
    def check(reaction, reactor):
        return (
            str(reaction.emoji) == "✅"
            and reaction.message.content == msg
            and reaction.message.channel.name == VERIFICATION_CHANNEL_NAME
            and any([r.name == MODERATOR_ROLE_NAME for r in
reactor.roles])
        )

    try:
        # Wait for a moderator to react with the correct emoji
        reaction, moderator = await bot.wait_for('reaction_add',
check=check)

        # Assign the "member" role to the user after verification
        member_role = discord.utils.get(guild.roles,
name=MEMBER_ROLE_NAME)
        if member_role is None:
            await ctx.send("Member role not found in the server.")
        else:
            await user.add_roles(member_role)
            await verification_channel.send(
                f"{user.mention} has been verified by {moderator.mention}
and given the {member_role.name} role."
            )

    except Exception as e:
        await verification_channel.send(f"An error occurred during the
verification process: {str(e)}")

# Run the bot
bot.run(TOKEN)
```

It was clear that the goal of the challenge was to get ourselves the `member` role in the discord server.

At first I was confused to as to what could possibly be done to get someone to click the verification symbol that had the moderator role. I read the documentation of the discord library and came to the conclusion that there wasn't anything worng with the role check itself. This mean't that it was something else.

At first I tried creating a thread with the name `verification` and it turns out that this allows us to bypass the channel check. This means that we can create an identical message in this thread and react to it with the checkmark. This will pass all the checks except the moderator role check, however the moderator role was the highest hurdle in the first place.

Upon closer inspection, I realized that the `check` function is called whenever the bot receives the `reaction_add` event. But the `bot` instance is not unique per server, as in the beginning of the `verifyme` function definition, the guild is acquired through the context:

```
async def request_verification(ctx):
    user = ctx.author
    guild = ctx.guild
```

Now looking back to the `check` function, we see that no checking is being done as to whether the reaction was from the same server or not.

```
    def check(reaction, reactor):
        return (
            str(reaction.emoji) == "✅"
            and reaction.message.content == msg
            and reaction.message.channel.name == VERIFICATION_CHANNEL_NAME
            and any([r.name == MODERATOR_ROLE_NAME for r in
reactor.roles])
        )
```

This means that we can invite the bot to a server where we are an administrator, create the `moderator` role and give it to ourselves, create a channel named `verification` and react to an identical message with the checkmark there, which will trigger the verification code to run in the main discord.

So, to obtain the flag:

1. Create a discord server
2. Invite the bot to the server (the one in the challenge discord, not one you create yourself)
3. Create the `moderator` role and the `verification` channel and assign the `moderator` role to yourself.
4. Send a `!verifyme` in the challenge discord
5. Create an identical message that was sent to the `verification` channel in the challenge server, in your own server's `verification` channel. Note that you need to copy the `moderator` roles ID inside the challenge server and replace the @mention with that inside of the message in your own, which results in the message being displayed as `unkown role`, but that is fine, since all that matter's is that the message is identical. This part was what was probably the issue why many could not solve this challenge, as the role ID was wrong, resulting in differing messages and not passing the `check` function.
6. React to the message with a checkmark emoji to obtain the flag