```
import discord
from discord.ext import commands
import random
import asyncio
import os           # To remove files and maybe something else
from PIL import Image          #for image processing and conversion
from PIL import ImageOps        #for inverted images
import requests     #download Discord Avatars
#import SH_gameclass

#pics_used = ['std','png']
pics_used = ['HD','jpg']
bot_id = 714561396851081257
voice_channel = 720281873825398784
sound_lib =
['Meinen_neuen_Lederlappen.mp3','menschlicher_Lappen1.mp3','menschlicher
_Lappen2.mp3','direkt_20_Stueck_geholt.mp3']
sound_fas =
['Schlurp1.mp3','Schlurp2.mp3','Schlurp3.mp3','Schlurp4.mp3','Schlurp6.m
p3','Schlurp7.mp3']
sound_lib_victory = ['UDSSR_shittyflute.mp3']
sound_hitler_dead = ['Der_zweite_Weltkrieg_macht_keinen_Spass_mehr.mp3']
sound_fas_victory =
['Id_like_to_sign_this_bill.mp3','How_is_that_a_victory.mp3']
sound_hitler_elected = ['Ich_kapituliere_nicht.mp3']
sound_crazy_dude =
['Crazydude2_low.mp3','Crazydude2.mp3','Crazydude.mp3']


def rescale_height(im1,im2):
    im2 =
im2.resize((int(float(im2.size[0])*float(im1.size[1]/float(im2.size[1]))
) ,im1.size[1]))
    return im2

def rescale_width(im1,im2):
    im2 =
im2.resize((im1.size[0],int(float(im2.size[1])*float(im1.size[0]/float(i
m2.size[0])))))
    return im2

def get_concat_h(im1, im2):
    dst = Image.new('RGB', (im1.width + im2.width, im1.height))
    dst.paste(im1, (0, 0))
    dst.paste(im2, (im1.width, 0))
    return dst


def get_concat_v(im1, im2):
    dst = Image.new('RGB', (im1.width, im1.height + im2.height))
    dst.paste(im1, (0, 0))
    dst.paste(im2, (0, im1.height))
    return dst


```

```
class SH(commands.Cog):
    #Class which contains the actuall game
    class SH_game:


        def __init__(self, players):
            presidential_powers =
[['None','None','Examine','Kill','Kill','None'],
['None','Identity','President','Kill','Kill','None'],
['Identity','Identity','President','Kill','Kill','None']]
            fs_nr = 1 + int((len(players)-5)/2)
            lib_nr = 3 + int((len(players)-4)/2)
            roles = ['hitler'] + ['fascist']*fs_nr + ['liberal']*lib_nr
            random.shuffle(players)
            random.shuffle(roles)
            self._players = dict(zip(players, roles))
            random.shuffle(players)
            self._player_order = list(players)
            self._next_presidents = players
            self._presidential_powers = presidential_powers[fs_nr -1]


            #ingame variables
            self._passed_policies = [0,0]
            #self._passed_policies = [0,0]
            self._draw_pile = ['fascist']*11 + ['liberal']*6
            random.shuffle(self._draw_pile)
            self._discard_pile = list()
            self._rejected_governments = 0
            self._nominated_chancellor = None
            self._last_government = [None]*2
            self._investigated = list()

        def return_party(self,player):
            if self._players[player] == 'liberal':
                return 'liberal'
            else:
                return 'fascist'

        def return_president(self):
            return self._next_presidents[0]

        def return_chancellor(self):
            return self._nominated_chancellor

        def return_players(self):
            return self._players.keys()

        def return_presidential_power(self):
            return self._presidential_powers[self._passed_policies[1]-1]

        def return_investigate_candidates(self):
            res = list()
            for x in self._players.keys():
                if x in self._investigated:
```

```
111                 continue
112             if is_president(x):
113                 continue
114             res.append(x)
115         return res
116 
117     def return_other_players(self):
118         res = list()
119         for x in self._players.keys():
120             if x != self._next_presidents[0]:
121                 res.append(x)
122         return res
123 
124     def investigate(self,player):
125         self._investigated.append(player)
126         return self.return_party(player)
127 
128 
129     def is_hitler(self,player):
130         if self._players[player] == 'hitler':
131             return True
132         else:
133             return False
134 
135 
136     def return_chancellor_candidates(self):
137         res = list(self._players.keys())
138         res.remove(self._next_presidents[0])
139         if self._last_government[1] == None:
140             return res
141         else:
142             if self._last_government[1] != self._next_presidents[0]:
143                 try:
144                     res.remove(self._last_government[1])
145                 except:
146                     pass
147             if(len(self._players.keys()) > 5):
148                 try:
149                     res.remove(self._last_government[0])
150                 except:
151                     pass
152         return res
153 
154 
155     def list_next_presidents(self):
156        return self._next_presidents
157 
158 
159     def is_president(self, player):
160         if self._next_presidents[0] == player:
161             return True
162         else:
163             return False
164 
165     def is_chancellor(self, player):
```

```
166             if self._nominated_chancellor == player:
167                 return True
168             else:
169                 return False
170 
171         def is_fascist(self,player):
172             if self._players[player] == 'fascist':
173                 return True
174             else:
175                 return False
176 
177 
178         def was_not_in_last_government(self,player):
179             return player in self._last_government
180 
181 
182         def reshuffle_deck(self):
183             self._draw_pile += self._discard_pile
184             random.shuffle(self._draw_pile)
185             self._discard_pile = list()
186 
187         #return all fascists and hitler
188         def show_fascists(self):
189             res = [list(),list()]
190             for x in self._players.keys():
191                 if (self._players[x] == 'fascist'):
192                     res[0].append(x)
193             for x in self._players.keys():
194                 if (self._players[x] == 'hitler'):
195                     res[1].append(x)
196             return res
197 
198 
199         def nominate_chancellor(self,player):
200             self._nominated_chancellor = player
201 
202 
203         def discard_card(self,card):
204             self._discard_pile.append(card)
205 
206 
207         def fascist_policy_victory(self):
208             if self._passed_policies[1] >= 6:
209                 return True
210             else:
211                 return False
212 
213 
214         def liberal_policy_victory(self):
215             if self._passed_policies[0] >= 5:
216                 return True
217             else:
218                 return False
219 
220 
```

```
221         def pass_policy(self,policy):
222             if (policy == 'fascist'):
223                 self._passed_policies[1] += 1
224             elif (policy == 'liberal'):
225                 self._passed_policies[0] += 1
226 
227 
228         #Insert the vote as a list
229         def enter_vote(self, vote):
230             hitler_won = False
231             if(sum(vote) > len(self._players)/2):
232                 if self.is_hitler(self._nominated_chancellor) and
233 self._passed_policies[1] >= 3:
234                     hitler_won = True
235                 self.make_chancellor()
236                 return(True, hitler_won,False)
237             else:
238                 policy_auto_passed = self.reject_chancellor()
239                 return(False,False,policy_auto_passed)
240 
241 
242         #make a new player chancellor
243         def make_chancellor(self):
244             self._rejected_governments = 0
245             self._last_government[0] = self.return_president()
246             self._last_government[1] = self.return_chancellor()
247 
248 
249         def change_government(self):
250             self._nominated_chancellor = None
251             last = self._next_presidents[0]
252             self._next_presidents.remove(last)
253             if not last in self._next_presidents: # Bc of Presidential
254 Power
255                 self._next_presidents.append(last)
256 
257 
258         #make a new player chancellor
259         def reject_chancellor(self):
260             self.change_government()
261             self._rejected_governments += 1
262             if (self._rejected_governments >= 3):
263                 self._rejected_governments = 0
264                 if(len(self._draw_pile) < 3):
265                     self.reshuffle_deck()
266                 x = self._draw_pile.pop(0)
267                 self.pass_policy(x)
268                 return True
269             return False
270 
271 
272         #draw policies and return the policies and the president
273         def draw_policies(self):
274             if(len(self._draw_pile) < 3):
275                 self.reshuffle_deck()
```

```
            drawn_policies =
[self._draw_pile.pop(0),self._draw_pile.pop(0),self._draw_pile.pop(0)]

            return (drawn_policies)


        def choose_president(self, player):
            self._next_presidents.insert(1,player)


        def examine_policies(self):
            if(len(self._draw_pile) < 3):
                self.reshuffle_deck()
            drawn_policies =
[self._draw_pile[0],self._draw_pile[1],self._draw_pile[2]]

            return (drawn_policies)

        def kill(self,player):
            hitler_lost = self.is_hitler(player)
            self._players.pop(player)
            self._next_presidents.remove(player)
            return hitler_lost


    def __init__(self, client):
        self.client = client
        self._lobby = None
        self._ballets = [None]
        print('SH starting')
        self._channel = self.client.get_channel(714576804899323984)
        self._voice_channel = self.client.get_channel(voice_channel)
        #global SH_player                # Currently makes this a global
variable, it works, but is ugly
        #SH_player = self._players

    @commands.Cog.listener()
    async def on_ready(self):
        print('SH is ready')

################################################################
#Checks to see if a player is allowed to execute commands   #
################################################################



    @commands.command(brief='Opns a new SH Game', description = 'Opens a
new Secret Hitler game by sending a message in the Secret Hitler chat,
which players can join by hitting the like button')
    async def open_SH(self,ctx):
        if (self._lobby == None):
            self._players = list()
            self._status = 'Waiting for Players'
            self._lobby = await self._channel.send('Waiting for players
to join Secret Hitler')
```

```
331             await self._lobby.add_reaction( '👍')
332         else:
333             await ctx.send('Already a Lobby running')
334 
335 
336     def is_player(self, player):
337         if(player in self._players):
338             return True
339         else:
340             return False
341 
342 
343     #Invert the result of a check for discord.py
344     def inverse_check(f):
345         def predicate(ctx):
346             return (not bool(f(ctx)))
347         return commands.check(predicate)
348 
349 
350     #Check if the Bot send the reaction
351     def check_if_it_is_bot(self, user):
352         return user.id != bot_id
353 
354 
355 #    #Check if the bot is doing a certain activity
356 #    def is_bot_activity(activity):
357 #        global bot_member
358 #        def predicate(ctx):
359 #            return bot_member.activity == activity
360 #        return commands.check(predicate)
361 
362 ##################################################################
363 #Admin and pregame Commands                                      #
364 ##################################################################
365 
366     async def remove_reaction(reation,user):
367         await reaction.remove(user)
368 
369 
370     @commands.Cog.listener()
371     async def on_reaction_add(self,reaction,user):
372         if not self.check_if_it_is_bot(user):
373             return
374         elif (reaction.message.id == self._lobby.id):
375             if (len(self._players) >= 10):
376                 await self._channel.send(f'The Game is already full')
377             elif not self.is_player(user):
378                 #await self._channel.send(f'{user.name} has joined the
379 lobby for Secret Hitler')
380                 self._players.append(user)                      # Add
381 the Player to the list
382                 #await reaction.remove(user)
383         elif (reaction.message.id in self._ballets and self._vote[user]
384 == None):
385             if reaction.emoji ==  '👍':
```

```
                self._vote[user] = True
                if (self.check_votes_missing() == 0):
                    await self.cast_votes()
            elif reaction.emoji ==  '👎':
                self._vote[user] = False
                if (self.check_votes_missing() == 0):
                    await self.cast_votes()


    @commands.Cog.listener()
    async def on_reaction_remove(self, reaction,user):
        if self.check_if_it_is_bot(user):
            return
        if (reaction.message.id == self._lobby.id):
            if  self.is_player(user):
                #await self._channel.send(f'{user} left the lobby for
Secret Hitler')
                self._players.remove(user)                        #
Remove the Player to the list
        elif (reaction.message.id in self._ballets and self._vote[user]
!= None):
            if reaction.emoji ==  '👍' and self._vote[user] == True:
                self._vote[user] = None
            elif reaction.emoji ==  '👎' and self._vote[user] == False:
                self._vote[user] = None



    @commands.command(brief='Return the status of the current SH game',
description="Sends the status of the currently active Secret Hitler game
into this chat. \n if the game is awaiting player votes, it will say,
which player haven't voted yet")
    async def status_SH(self,ctx):
        await ctx.send(self._status)
        if (self._status ==  'Waiting for players to cast their vote'):
            res = 'Waiting on '
            for x in self._players:
                if(self._vote[x] == None):
                    res = res + x.name
            res +=  ' to cast their votes'
            await ctx.send(res)


    def check_vote(self,reaction,user,player,message):
        return user == player and reaction.message.id == message and
(reaction.emoji =='👍' or reaction.emoji == '👎')


    @commands.command(brief='Starts the game with the current players',
description = 'It starts an instance of Secret Hitler, with all players
which upvoted the game post. \n The bot will then remove the post, and
starts the game. \n It will send the roles out in privat, randomly
selects a player order.')
```

```
441    #@commands.check(Salo_has_Birthday)
442    @commands.has_any_role('Game_master')
443    async def start_SH(self, ctx):
444        if (self._status !=  'Waiting for Players' or self._lobby ==
445None):
446            await ctx.send(f'There is no game to start')
447#        elif (len(self._players) < 5):
448#            await ctx.send(f'The Game needs at least {5 -
449len(self._players)} more Players to start')
450        else:
451            print("Voice Channel joined")
452            self._vc = await self._voice_channel.connect()
453            self._nr_players = 5
454            #self._nr_players = len(self._players)
455            self._last_player = None
456            await self._lobby.delete()
457            await self.client.change_presence(status =
458discord.Status.online, activity=discord.Game('Secret Hitler'))
459            await self._channel.send('Secret Hitler is starting')
460            self._SH_game = self.SH_game(self._players)
461            for x in self._players:
462#Sending the fascist Info out
463                if self._SH_game.is_fascist(x):
464                    await self.SH_send_fascists(x)
465                if (len(self._players) < 7):
466                    if self._SH_game.is_hitler(x):
467                        await self.SH_send_fascists(x)        #Hitler
468gets fascist Info
469            for x in self._players:
470# Send out player roles
471                await x.send(file =
472discord.File(f'cogs/picture/{pics_used[0]}/{self._SH_game._players[x]}_R
473ole.{pics_used[1]}'))
474                await x.send(f'Your role is:
475{self._SH_game._players[x]}')
476            #Load the player Avatars
477            with requests.get(self._SH_game._player_order[0].avatar_url)
478as r:
479                img_data = r.content
480            with open('pics/image_name.webp', 'wb') as handler:
481                handler.write(img_data)
482            self._player_avatar =
483Image.open("pics/image_name.webp").convert("RGB")
484            for x in self._SH_game._player_order[1:]:
485                with requests.get(x.avatar_url) as r:
486                    img_data = r.content
487                with open('pics/image_name.webp', 'wb') as handler:
488                    handler.write(img_data)
489                im = Image.open("pics/image_name.webp").convert("RGB")
490                self._player_avatar = get_concat_h(self._player_avatar,
491im.resize((self._player_avatar.size[1],self._player_avatar.size[1])))
492            os.remove('pics/image_name.webp')
493            await self.SH_draw_board()
494            await self.SH_next_round()
495
```

```
496
497     #End this game instanz
498     async def end_game_SH(self):
499         await self.client.change_presence(status = discord.Status.idle,
500 activity=discord.Activity(name = 'Secret Hitler, waiting for players',
501 type = discord.ActivityType.watching))
502         await self._channel.send('Secret Hitler Game ended')
503         self._lobby = None      # Remove the lobby
504         self._SH_game = None
505         try:
506             await self._vc.disconnect()
507             os.remove('pics/order.jpeg')
508         except:
509             print('game canceled')
510
511     #Unloads and disables all SH commands
512     @commands.command(brief='Unloads the SH cog', description='Stops the
513 current game of Secret Hitler, and unloads its cog')
514     @commands.has_any_role('Game_master')
515     async def unload_SH(self, ctx):
516         await self.end_game_SH()
517         await self.client.change_presence(status = discord.Status.idle,
518 activity=discord.Game('waiting'))
519         await ctx.send('Secret Hitler Plugin unloaded')
520         print('SH Unloaded')
521         self.client.unload_extension(f'cogs.SH')
522
523 ##############################################################
524 # Ingame Commands: Should later only be enabled for players #
525 ##############################################################
526
527     async def SH_pick_player(self, eligiable_candidates, message, remove
528 = True):
529         candidates = [None] * len(eligiable_candidates)
530         candidates_id = [None] * len(eligiable_candidates)
531         for i in range (len(eligiable_candidates)):
532             candidates[i] = await
533 self._channel.send(message.format(player = eligiable_candidates[i]))
534             await candidates[i].add_reaction('👍')
535             candidates_id[i] = candidates[i].id
536         reaction, user = await
537 self.client.wait_for('reaction_add',check=lambda reaction,user:
538 self.president_pick_test(reaction,user,candidates_id))
539         pick = None
540         for i in range (len(candidates_id)):
541             if reaction.message.id == candidates_id[i]:
542                 pick = eligiable_candidates[i]
543                 if remove:
544                     await candidates[i].delete()
545             else:
546                 await candidates[i].delete()
547         return pick
548
549
550     #Made with VIM <3
```

```
551    async def SH_draw_players(self):
552        res = Image.new('RGB', (0,0))
553        for x in self._SH_game._player_order:
554            if self._SH_game.is_president(x):
555                img =
556Image.open(f'cogs/picture/{pics_used[0]}/president.jpg')
557                if res.size[0] == 0:
558                    res = img
559                else:
560                    res = get_concat_h(res, img)
561            elif self._SH_game.is_chancellor(x):
562                img =
563Image.open(f'cogs/picture/{pics_used[0]}/chancellor.jpg')
564                if res.size[0] == 0:
565                    res = img
566                else:
567                    res = get_concat_h(res, img)
568            elif x == self._SH_game._last_government[0]:
569                img =
570Image.open(f'cogs/picture/{pics_used[0]}/last_president.jpg')
571                if res.size[0] == 0:
572                    res = img
573                else:
574                    res = get_concat_h(res, img)
575            elif x == self._SH_game._last_government[1]:
576                img =
577Image.open(f'cogs/picture/{pics_used[0]}/last_chancellor.jpg')
578                if res.size[0] == 0:
579                    res = img
580                else:
581                    res = get_concat_h(res, img)
582            else:
583                img =
584Image.open(f'cogs/picture/{pics_used[0]}/empty.jpg')
585                if res.size[0] == 0:
586                    res = img
587                else:
588                    res = get_concat_h(res, img)
589        self._positions = rescale_height(Image.new('RGB',(0,50)), res)
590        res = get_concat_v(self._positions,
591rescale_width(self._positions, self._player_avatar))
592        res.save('pics/player_state.jpeg')
593        self._last_player = await self._channel.send(file =
594discord.File('pics/player_state.jpeg'))
595        os.remove('pics/player_state.jpeg')
596
597
598    async def SH_draw_board(self):
599        lib_passed = self._SH_game._passed_policies[0]
600        fas_passed = self._SH_game._passed_policies[1]
601        gov_rej = self._SH_game._rejected_governments
602        im_board_lib =
603Image.open(f'cogs/picture/{pics_used[0]}/{gov_rej+1}Liberal_board{lib_pa
604ssed}.{pics_used[1]}')
605        im_board_fas =
```

```
606 Image.open(f'cogs/picture/{pics_used[0]}/{self._nr_players}fascist_board
607 {fas_passed}.{pics_used[1]}').resize(im_board_lib.size)
608         im_draw =
609 Image.open(f'cogs/picture/{pics_used[0]}/Policy{len(self._SH_game._draw_
610 pile)}.{pics_used[1]}')
611         im_discard =
612 Image.open(f'cogs/picture/{pics_used[0]}/Policy{len(self._SH_game._disca
613 rd_pile)}.{pics_used[1]}')
614         im_draw = rescale_height(im_board_lib, im_draw)
615         im_discard = rescale_height(im_board_lib,
616 ImageOps.invert(im_discard))
617         res_board =
618 get_concat_v(get_concat_h(im_board_lib,im_draw),get_concat_h(im_board_fa
619 s,im_discard))
620         res_board.save('pics/board_state.jpeg')
621         await self._channel.send(file =
622 discord.File('pics/board_state.jpeg'))
623         os.remove('pics/board_state.jpeg')
624         await self.SH_draw_players()
625
626         if self._SH_game.fascist_policy_victory():
627             self._vc.play(discord.FFmpegPCMAudio(f'cogs/Meme/{random.cho
628 ice(sound_fas_victory)}'))
629             while self._vc.is_playing():
630                 await asyncio.sleep(10)
631             await self._channel.send('Fascists won through signing
632 fascist laws')
633             await self.end_game_SH()
634         elif self._SH_game.liberal_policy_victory():
635             self._vc.play(discord.FFmpegPCMAudio(f'cogs/Meme/{random.cho
636 ice(sound_lib_victory)}'))
637             while self._vc.is_playing():
638                 await asyncio.sleep(10)
639             await self._channel.send('Liberals won through signing
640 liberal laws')
641             await self.end_game_SH()
642
643
644     async def SH_next_round(self):
645         if self._last_player != None:
646             await self._last_player.delete()
647         await self.SH_draw_players()
648         #await self.SH_draw_board()
649         self._status = 'Waiting for a chancelor to be announced'
650         await self.SH_make_chancellor()
651         chancellor =  self._SH_game.return_chancellor()
652         self._vote = dict(zip(self._SH_game.return_players(),
653 [None]*len(self._SH_game.return_players())))
654         await self._last_player.delete()
655         await self.SH_draw_players()
656         self._status = 'Waiting for players to cast their vote'
657         await
658 self.SH_start_vote(self._SH_game.return_president().name,chancellor.name
659 )
660
```

```
661    async def Legeslative_action(self):
662        policies = await self.SH_draw_policies()
663        president = self._SH_game.return_president()
664        chancellor = self._SH_game.return_chancellor()
665        policies = await
666self.SH_president_select_policies(president,policies)
667        self._status = 'Policies are being passed on to the chancellor'
668        policy = await self.SH_chancellor_select_policies(president,
669chancellor, policies)
670        await self.SH_draw_board()
671        if policy == 'fascist':
672            self._vc.play(discord.FFmpegPCMAudio(f'cogs/Meme/{random.cho
673ice(sound_fas)}'))
674            if self._SH_game._passed_policies[0] == 3 and
675self._SH_game._passed_policies[1]:
676                self._vc.play(discord.FFmpegPCMAudio('cogs/Meme/{random.
677choice(sound_crazy_dude)}'))
678            lib_won = await self.SH_presidential_power()
679            if lib_won:
680                await self.end_game_SH()
681                return
682        if policy == 'liberal':
683            self._vc.play(discord.FFmpegPCMAudio(f'cogs/Meme/{random.cho
684ice(sound_lib)}'))
685
686        self._SH_game.change_government()
687        await self.SH_next_round()
688
689    @commands.command(brief='Shows the other fascists',
690description='This command only works when you are fascist in the active
691game.\n It sends the player names of the other fascists to the author in
692privat.')
693    async def SH_show_fascists(self,ctx):
694        if( self._SH_game.is_fascist(ctx.author)):
695            SH_send_fascists(self,ctx.author)
696        else:
697            await player.send('Damn you cheeky liberals')
698
699
700    async def SH_send_fascists(self, player):
701            fascists = self._SH_game.show_fascists()
702            res = 'Fasicsts are: '
703            for x in fascists[0]:
704                res += x.name
705            res += '\nHitler: '
706            res += fascists[1][0].name
707            await player.send(res)
708
709
710    @commands.command(brief='lists the next presidents',
711description='Returns a string with the order of next presidents. \n This
712string is send into the chat, in which the command is executed.')
713    async def SH_next_presidents(self,ctx):
714        res = 'The next presidents are: '
715        for x in self._SH_game.list_next_presidents():
```

```
716             res += str(x.name) + ', '
717         await ctx.send(res)
718 
719 
720     def president_pick_test(self, reaction, user, candidates):
721         return self._SH_game.is_president(user) and reaction.message.id
722 in candidates and reaction.emoji =='👍'
723 
724     async def SH_make_chancellor(self):
725         chancellor = await
726 self.SH_pick_player(self._SH_game.return_chancellor_candidates(),
727 "Nominate **{player.name}** as the next chancellor")
728         self._SH_game.nominate_chancellor(chancellor)
729 
730 
731     async def SH_start_vote(self, president, chancellor):
732         self._ballets = [None] * len(self._players)
733         for i in range (len(self._players)):
734             self._ballets[i] = await self._players[i].send(f'Please
735 enter your Vote for this Government: \nPresident:
736 **{president}**\nChancellor: **{chancellor}**')
737             await self._ballets[i].add_reaction('👍')
738             await self._ballets[i].add_reaction('👎')
739             self._ballets[i] = self._ballets[i].id
740 
741 
742     def check_votes_missing(self):
743         count = 0
744         for x in self._SH_game._players.keys():
745             if self._vote[x] == None:
746                 count += 1
747         return count
748 
749     async def cast_votes(self):
750         self._ballets = [None]
751         await self._last_player.delete()
752 
753         voted = self._player_avatar.copy()
754         yes = Image.open(f'cogs/picture/{pics_used[0]}/yes.png')
755         yes.convert('RGBA')
756         no = Image.open(f'cogs/picture/{pics_used[0]}/no.png')
757         no.convert('RGBA')
758         height = voted.size[1]
759         yes = rescale_height(self._player_avatar,yes)
760         no = rescale_height(self._player_avatar,no)
761         for x in self._SH_game._players.keys():
762             if self._vote[x] == True:
763                 voted.paste(yes,
764 (self._SH_game._player_order.index(x)*height,0),yes)
765             else:
766                 voted.paste(no,
767 (self._SH_game._player_order.index(x)*height,0),no)
768         voted.resize(self._player_avatar.size)
769         res = get_concat_v(self._positions,
770 
```

```
771 rescale_width(self._positions, voted))
772         res.save('pics/voted.jpeg')
773         self._last_player = await self._channel.send(file =
774 discord.File('pics/voted.jpeg'))
775         os.remove('pics/voted.jpeg')
776 
777         president = self._SH_game.return_president()
778         chancellor = self._SH_game.return_chancellor()
779         passed, hitler_won, auto_passed =
780 self._SH_game.enter_vote(self._vote.values())
781         if (passed):
782             if (hitler_won):
783                 await self._channel.send('You have elected Hitler as
784 chancellor past 1933, the fascists have won')
785                 await self.end_game_SH()
786                 return
787             self._status = ' Passing policies'
788             await self.Legeslative_action()
789         else:
790             await self.SH_draw_board()
791             await self.SH_next_round()
792 
793 
794     async def SH_president_select_policies(self,president, policies):
795         policy_cards = list()
796         for x in policies:
797             if x == 'liberal':
798                 policy_cards.append(discord.File(f'cogs/picture/{pics_us
799 ed[0]}/liberal.{pics_used[1]}'))
800             elif x == 'fascist':
801                 policy_cards.append(discord.File(f'cogs/picture/{pics_us
802 ed[0]}/fascist.{pics_used[1]}'))
803         policy_message = [None]*3
804         for i in range (len(policies)):
805             policy_message[i] = await
806 president.send(file=policy_cards[i])
807             await policy_message[i].add_reaction('👍')
808             await policy_message[i].add_reaction('👎')
809 
810         passed_policies = [None]*3
811         while True:
812             for i in range (3):
813                 reaction, user = await
814 self.client.wait_for('reaction_add',check=lambda reaction,user:
815 self.check_vote(reaction,user,president,policy_message[i].id))
816                 #await reaction.message.clear_reactions()
817                 #await president.send(reaction.emoji)
818                 if (reaction.emoji == '👍'):
819                     passed_policies[i] = True
820                 elif (reaction.emoji == '👎'):
821                     passed_policies[i] = False
822             if(sum(passed_policies) == 2):
823                 break
824             else:
825                 await president.send('Invalid vote, all votes have to be
```

```
recasted')
        await president.send('Vote is correctly selected')
        for i in range (3):
            if passed_policies[i] == False:
                self._SH_game.discard_card(policies[i])
                policies.pop(i)
        return (policies)


    async def SH_chancellor_select_policies(self,president,chancellor,
policies):
        policy_cards = list()
        for x in policies:
            if x == 'liberal':
                policy_cards.append(discord.File(f'cogs/picture/{pics_us
ed[0]}/liberal.{pics_used[1]}'))
            elif x == 'fascist':
                policy_cards.append(discord.File(f'cogs/picture/{pics_us
ed[0]}/fascist.{pics_used[1]}'))

        policy_message = [None]*2
        for i in range (2):
            policy_message[i] = await
chancellor.send(file=policy_cards[i])
            await policy_message[i].add_reaction('👍')
            await policy_message[i].add_reaction('👎')

        passed_policies = [None]*2
        if await self.SH_veto(president,chancellor,policies):
            return
        while True:
            for i in range (2):
                reaction, user = await
   self.client.wait_for('reaction_add',check=lambda reaction,user:
   self.check_vote(reaction,user,chancellor,policy_message[i].id))
                #await reaction.message.clear_reactions()
                #await chancellor.send(reaction.emoji)
                if (reaction.emoji == '👍'):
                    passed_policies[i] = True
                elif (reaction.emoji == '👎'):
                    passed_policies[i] = False
            if(sum(passed_policies) == 1):
                break
            else:
                await chancellor.send('Invalid vote, all votes have to
   be recasted')
        await chancellor.send('Voted is correctly selected')
        res = None
        for i in range (2):
            if passed_policies[i] == False:
                self._SH_game.discard_card(policies[i])
            elif passed_policies[i] == True:
                res = policies[i]
                self._SH_game.pass_policy(policies[i])
        return res
```

```
    async def SH_draw_policies(self):
        self._status = 'The policies are being send out to our
president'
        return self._SH_game.draw_policies()


    async def SH_investigate(self,president):
        member = await
self.SH_pick_player(self._SH_game.return_other_players(), "Investigate
**{player.name}**s party membership",False)
        await president.send(file =
discord.File(f'cogs/picture/{pics_used[0]}/{self._SH_game.return_party(m
ember)}_Party.{pics_used[1]}'))
        await president.send(f'**{member.name}** is
**{self._SH_game.return_party(member)}**.')
        return member

    async def SH_make_next_president(self):
        member = await
self.SH_pick_player(self._SH_game.return_other_players(), "Make
**{player.name}** the next President")
        res = None
        for x in self.client.get_all_members():
            if x.name == member.name and x.discriminator ==
member.discriminator:
                res = x
        self._SH_game.choose_president(res)
        return member


    async def SH_execution(self):
        member = await
self.SH_pick_player(self._SH_game.return_other_players(), "Kill
**{player.name}**")
        img = Image.open(f'cogs/picture/{pics_used[0]}/skull.png')
        img = rescale_height(self._player_avatar, img)
        for i in range (len(self._SH_game._player_order)):
            if member == self._SH_game._player_order[i]:
                self._player_avatar.paste(img,
(i*self._player_avatar.size[1],0),img)
        hitler_lost = self._SH_game.kill(member)
        return member, hitler_lost


    async def SH_presidential_power(self):
        president = self._SH_game.return_president()
        if self._SH_game.return_presidential_power() == 'None':
            return False
        elif self._SH_game.return_presidential_power() == 'Examine':
            await self._channel.send('The President can inspect the next
3 policies')
            policies = self._SH_game.examine_policies()
            for x in policies:
```

```
                if x == 'liberal':
                    await president.send(file =
discord.File(f'cogs/picture/{pics_used[0]}/liberal.{pics_used[1]}'))
                elif x == 'fascist':
                    await president.send(file =
discord.File(f'cogs/picture/{pics_used[0]}/fascist.{pics_used[1]}'))
            return
        elif self._SH_game.return_presidential_power() == 'Identity':
            await self._channel.send('The President can investigate
another players Identity')
            player = await
self.SH_investigate(self._SH_game.return_president())
            await self._channel.send(f'{president.name} investigated
{player.name} party membership')
        elif self._SH_game.return_presidential_power() == 'President':
            await self._channel.send('The President can choose the next
presidential candidate')
            player = await self.SH_make_next_president()
            await self._channel.send(f'{president.name} chose
{player.name} as the next president candidate')
        elif self._SH_game.return_presidential_power() == 'Kill':
            #await self._channel.send('The President Must kill a
player')
            member, hitler_lost = await self.SH_execution()
            await self._channel.send(f'{president.name} killed
{member.name}')
            self._vc.play(discord.FFmpegPCMAudio('cogs/Meme/coffin_dance
.mp3'))
            if hitler_lost:
                await self._channel.send('The liberals won, by killing
Hitler!')
                return True
        return False


    async def SH_veto(self,president,chancellor,policies):
        if self._SH_game._passed_policies[1] == 5:
            veto = await chancellor.send('The Veto power has been
unlocked, do you want to reject these policies?')
            await veto.add_reaction('👍')
            await veto.add_reaction('👎')
            reaction, user = await self.client.wait_for('reaction_add',
check=lambda reaction,user:
self.check_vote(reaction,user,chancellor,veto.id))
            if (reaction.emoji == '👍'):
                await self._channel.send('The Chancellor requested
Veto')
                veto_check = await president.send('The Chancellor used
his veto, do you accept it?')
                await veto_check.add_reaction('👍')
                await veto_check.add_reaction('👎')
                reaction, user = await
self.client.wait_for('reaction_add', check=lambda reaction,user:
self.check_vote(reaction,user,president,veto_check.id))
```

```
                if (reaction.emoji == '👍'):
                    for x in policies:
                        self._SH_game.discard_card(x)
                    return True
                await self._channel.send('The President rejected the
Veto')
                await chancellor.send('The President rejected your Veto,
please cast your vote.')
        return False




def setup(client):
    client.add_cog(SH(client))
```