Python

Developing KakaoTalk automation with Python

Progress Report : 2

Date : 2023/12/10

Name : Dong Ho Han

ID : 203734

1. Introduction

1) Background

While looking at examples for automation, I often saw automation systems that send emails, but I did not see any cases of automating KakaoTalk, which is closest to our everyday life. I thought that if it was well-made, it would be much more valuable than code that automatically sends emails, so I chose this topic.

2) Project goal

To enable automatic sending of pre-written messages to specific friend lists..

3) Differences from existing programs

When I looked at some automatic email sending programs, the processes of launching the program and logging in were not automated. In this project, I plan to automate the process from launching the program to closing the program.

2. Functional Requirement

1) Function 1

- Auto-launch and login (login information is included in the code in advance)

(1) Detailed function

- Add a function to close and launch KakaoTalk for initialization

2) Function 2

- Friend search function – Find friends that are pre-entered in the code

-

3) Function 3

- Content input and transmission – Send messages to friends by automating keyboard and mouse controls

(1) Detailed function

- Add a function to close the chat window after sending a message

3. Progress

1) Function implementation

1) Function 1

- Auto-launch and login (login information is included in the code in advance)

(1) Detailed function

- Add a function to close and launch KakaoTalk for initialization

2) Function 2

- Friend search function – Find friends that are pre-entered in the code

3) Function 3

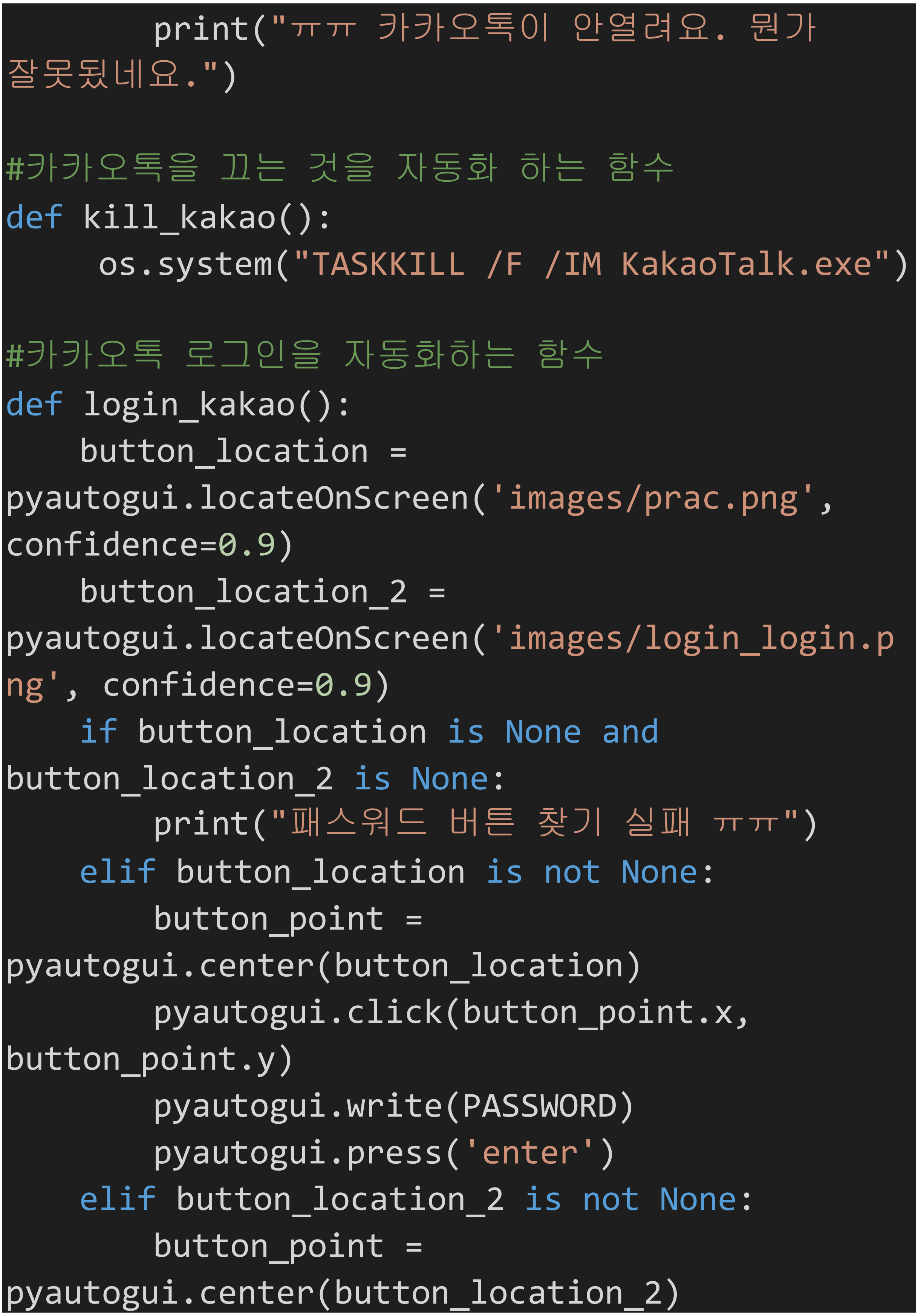
- Content input and transmission – Send messages to friends by automating keyboard and mouse controls

(1) Detailed function

- Add a function to close the chat window after sending a message

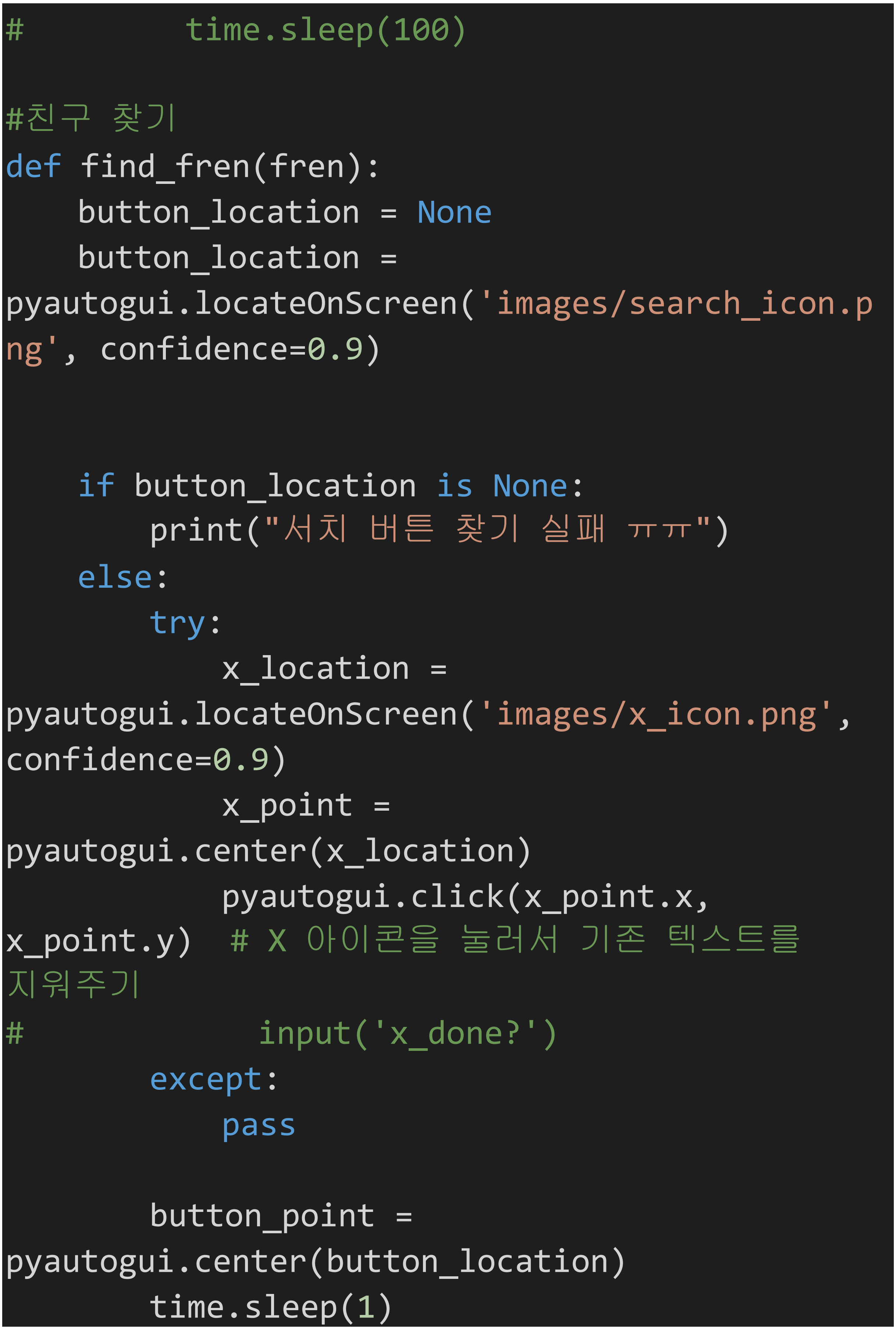
I have implemented most of the features using Python. However, there were some errors during this process. While there is no issue with launching KakaoTalk, there is a problem with login recognition due to variations in computer speeds. As of now, these issues remain unresolved, and I am submitting the code with only the implementation of most functionalities. I will make efforts to address these issues before the final submission.

```python
#필요할법한 모듈들을 긁어 모아봤습니다.
import cv2 as cv
from difflib import get_close_matches
import subprocess
import os
import time
import pyautogui
import pyperclip

#아이디와 비밀번호를 입력받습니다.
ID = input("아이디를 입력하세요 : ")
PASSWORD = input("비밀번호를 입력하세요 : ")

#카카오톡을 키는 것을 자동화 하는 함수
def open_kakao():
    try:
        path = r"C:\Program Files (x86)\Kakao\KakaoTalk\KakaoTalk.exe"
        subprocess.Popen(path)
        print('카카오톡을 열어볼까용~!')
    except:
```

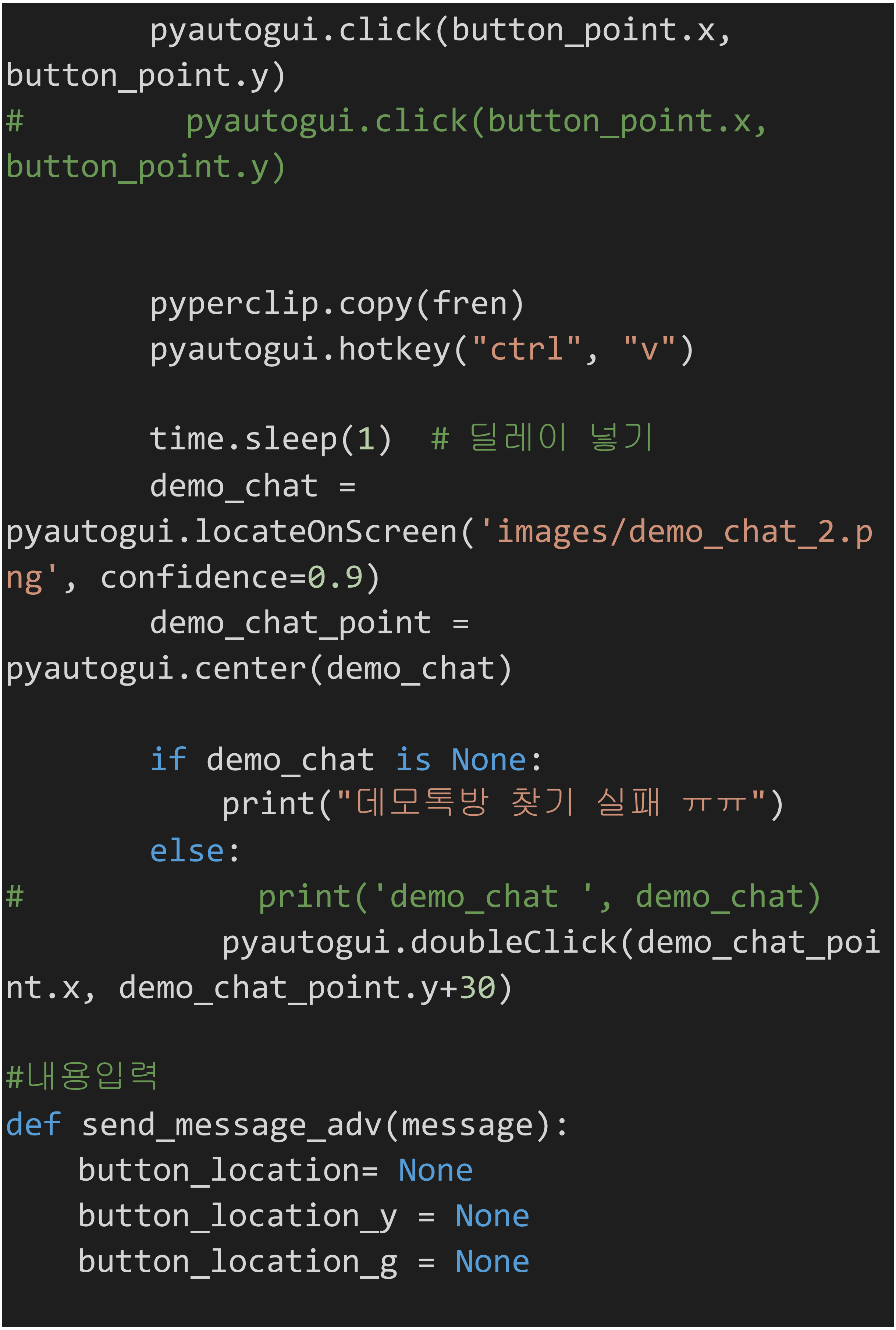


```
        print("ㅠㅠ 카카오톡이 안열려요. 뭔가
잘못됬네요.")

#카카오톡을 끄는 것을 자동화 하는 함수
def kill_kakao():
     os.system("TASKKILL /F /IM KakaoTalk.exe")

#카카오톡 로그인을 자동화하는 함수
def login_kakao():
    button_location =
pyautogui.locateOnScreen('images/prac.png',
confidence=0.9)
    button_location_2 =
pyautogui.locateOnScreen('images/login_login.p
ng', confidence=0.9)
    if button_location is None and
button_location_2 is None:
        print("패스워드 버튼 찾기 실패 ㅠㅠ")
    elif button_location is not None:
        button_point =
pyautogui.center(button_location)
        pyautogui.click(button_point.x,
button_point.y)
        pyautogui.write(PASSWORD)
        pyautogui.press('enter')
    elif button_location_2 is not None:
        button_point =
pyautogui.center(button_location_2)
```

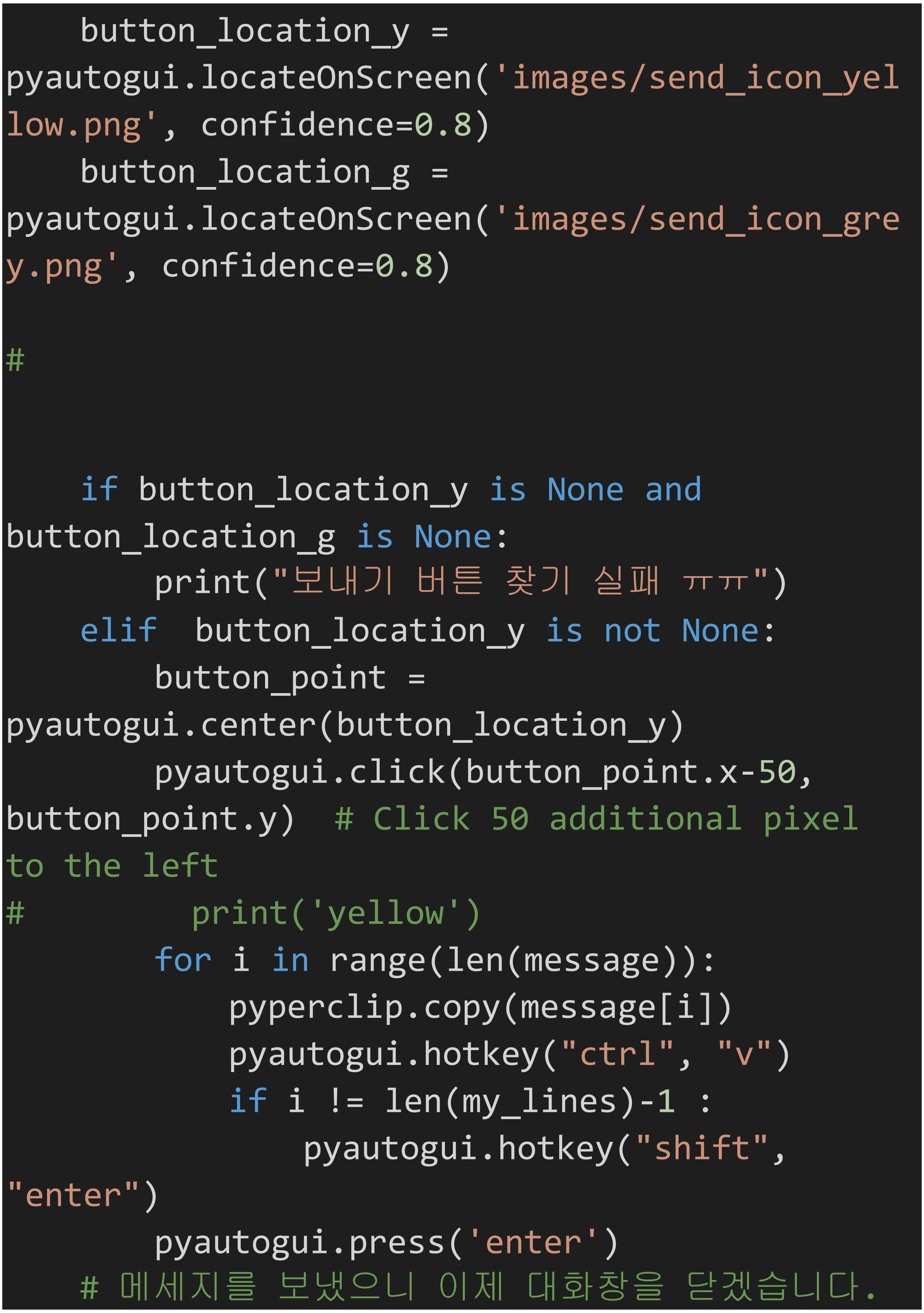
```
#     pyautogui.moveTo(button_point.x,
button_point.y, duration=0)
#     pyautogui.move(0, -45, 0.5,
pyautogui.easeInQuad)  # Move 45 pixels Up
        pyautogui.doubleClick(button_point.x,
button_point.y-45)
        pyautogui.write(PASSWORD)
        pyautogui.press('enter')

login_result = login_kakao()
print(login_result)

if login_result == 'success':
    pass
elif login_result == 'fail':


    counter = 0
    while counter <5 and login_result  ==
'fail':
        counter += 1
        print('1분 후 다시 시도합니다')

        time.sleep(60)
        print('attempt : ', counter)

        login_result = login_kakao()
        if login_result == 'success':
            print('로그인 성공')
```



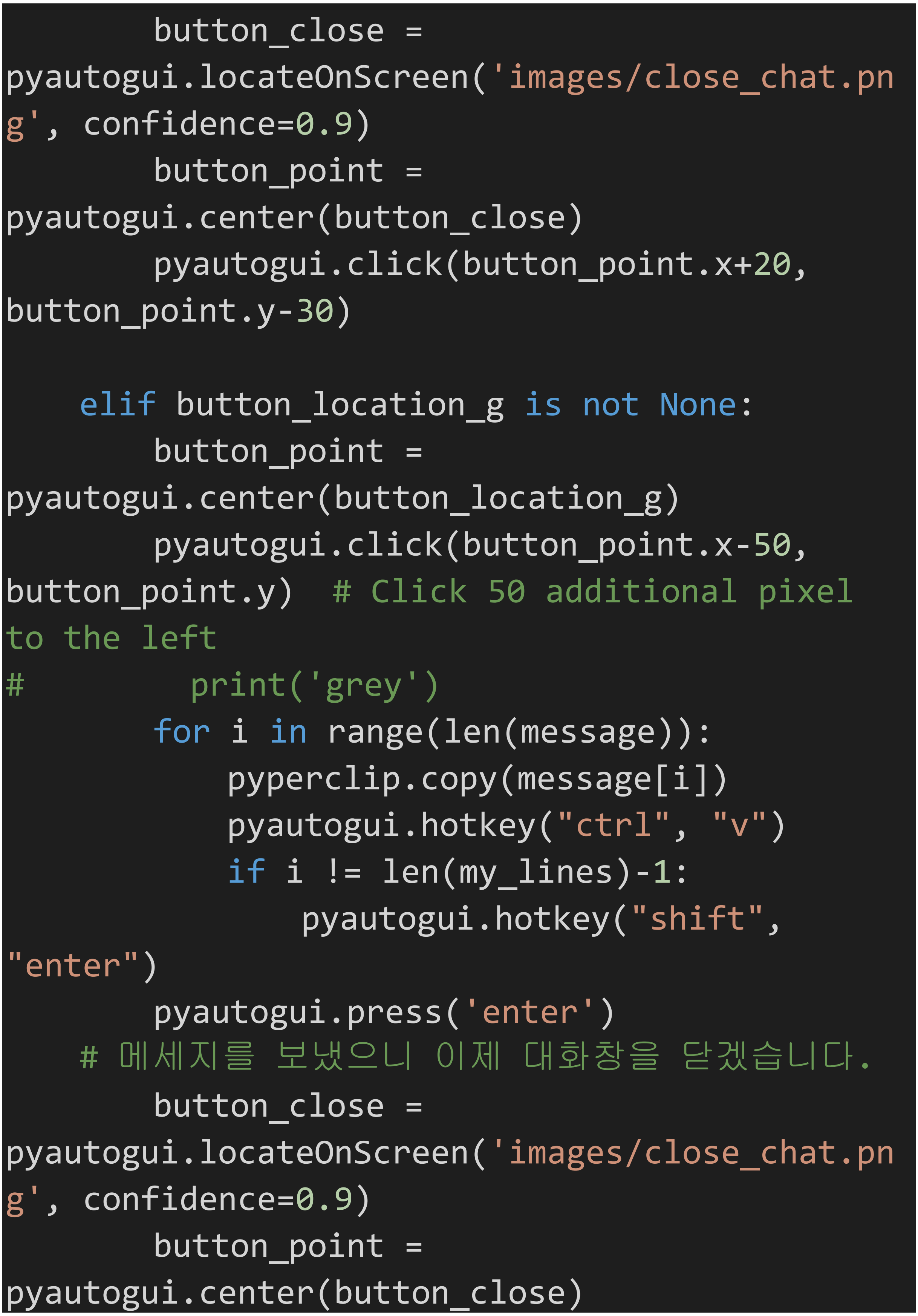
```python
#         time.sleep(100)

#친구 찾기
def find_fren(fren):
    button_location = None
    button_location = pyautogui.locateOnScreen('images/search_icon.png', confidence=0.9)


    if button_location is None:
        print("서치 버튼 찾기 실패 ㅠㅠ")
    else:
        try:
            x_location = pyautogui.locateOnScreen('images/x_icon.png', confidence=0.9)
            x_point = pyautogui.center(x_location)
            pyautogui.click(x_point.x, x_point.y)  # X 아이콘을 눌러서 기존 텍스트를 지워주기
#             input('x_done?')
        except:
            pass

        button_point = pyautogui.center(button_location)
        time.sleep(1)
```



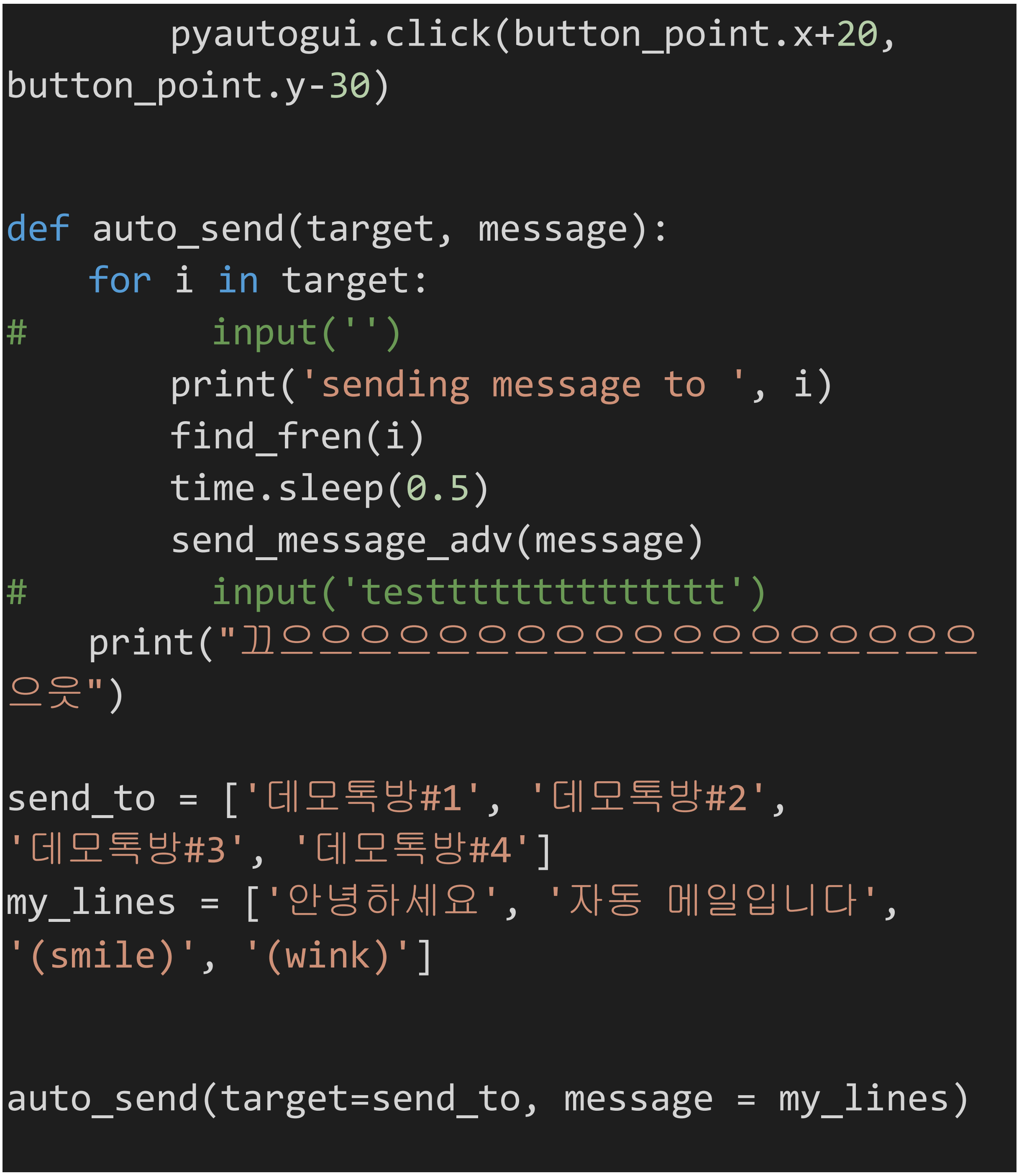
```
        pyautogui.click(button_point.x,
button_point.y)
#         pyautogui.click(button_point.x,
button_point.y)


        pyperclip.copy(fren)
        pyautogui.hotkey("ctrl", "v")

        time.sleep(1)  # 딜레이 넣기
        demo_chat =
pyautogui.locateOnScreen('images/demo_chat_2.p
ng', confidence=0.9)
        demo_chat_point =
pyautogui.center(demo_chat)

        if demo_chat is None:
            print("데모톡방 찾기 실패 ㅠㅠ")
        else:
#             print('demo_chat ', demo_chat)
            pyautogui.doubleClick(demo_chat_poi
nt.x, demo_chat_point.y+30)

#내용입력
def send_message_adv(message):
    button_location= None
    button_location_y = None
    button_location_g = None
```



```
    button_location_y = pyautogui.locateOnScreen('images/send_icon_yellow.png', confidence=0.8)
    button_location_g = pyautogui.locateOnScreen('images/send_icon_grey.png', confidence=0.8)

#


    if button_location_y is None and button_location_g is None:
        print("보내기 버튼 찾기 실패 ㅠㅠ")
    elif  button_location_y is not None:
        button_point = pyautogui.center(button_location_y)
        pyautogui.click(button_point.x-50, button_point.y)  # Click 50 additional pixel to the left
#         print('yellow')
        for i in range(len(message)):
            pyperclip.copy(message[i])
            pyautogui.hotkey("ctrl", "v")
            if i != len(my_lines)-1 :
                pyautogui.hotkey("shift", "enter")
        pyautogui.press('enter')
    # 메세지를 보냈으니 이제 대화창을 닫겠습니다.
```



```
        button_close = pyautogui.locateOnScreen('images/close_chat.png', confidence=0.9)
        button_point = pyautogui.center(button_close)
        pyautogui.click(button_point.x+20, button_point.y-30)

    elif button_location_g is not None:
        button_point = pyautogui.center(button_location_g)
        pyautogui.click(button_point.x-50, button_point.y)  # Click 50 additional pixel to the left
#         print('grey')
        for i in range(len(message)):
            pyperclip.copy(message[i])
            pyautogui.hotkey("ctrl", "v")
            if i != len(my_lines)-1:
                pyautogui.hotkey("shift", "enter")
        pyautogui.press('enter')
    # 메세지를 보냈으니 이제 대화창을 닫겠습니다.
        button_close = pyautogui.locateOnScreen('images/close_chat.png', confidence=0.9)
        button_point = pyautogui.center(button_close)
```

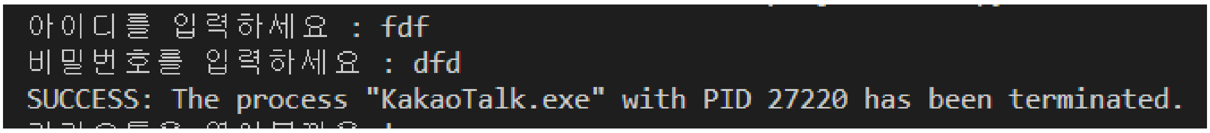


```
        pyautogui.click(button_point.x+20,
button_point.y-30)


def auto_send(target, message):
    for i in target:
#          input('')
        print('sending message to ', i)
        find_fren(i)
        time.sleep(0.5)
        send_message_adv(message)
#          input('testtttttttttttttt')
    print("끄으으으으으으으으으으으으으으으으으으
으읏")

send_to = ['데모톡방#1', '데모톡방#2',
'데모톡방#3', '데모톡방#4']
my_lines = ['안녕하세요', '자동 메일입니다',
'(smile)', '(wink)']


auto_send(target=send_to, message = my_lines)
```

2) Test results

(1) opening/closing kakao program

-

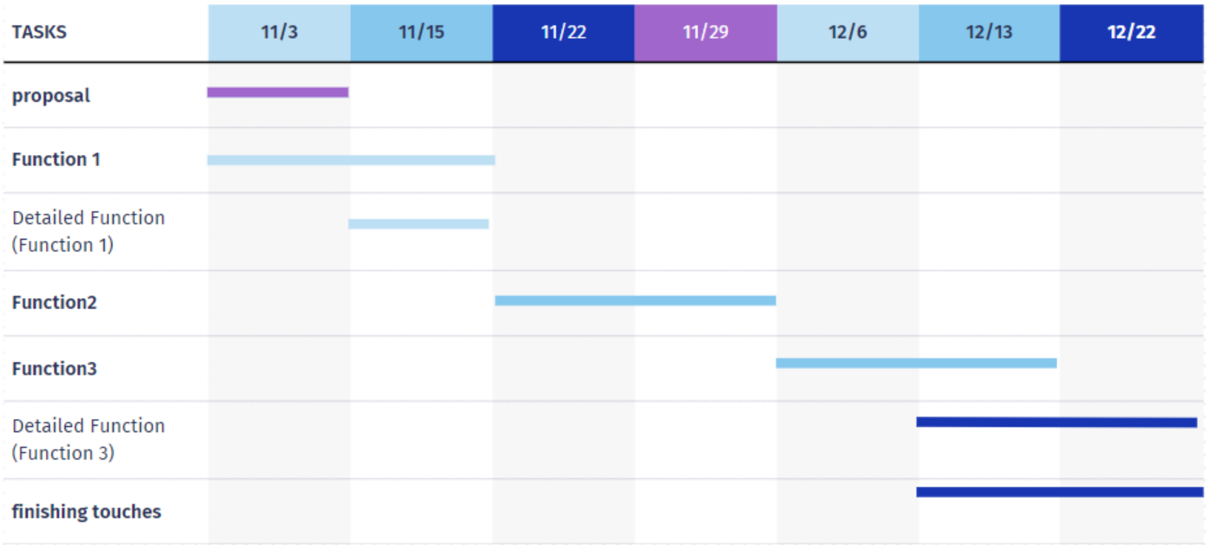


```
아이디를 입력하세요 : fdf
비밀번호를 입력하세요 : dfd
SUCCESS: The process "KakaoTalk.exe" with PID 27220 has been terminated.
```

4. Changes in Comparison to the Plan

Adjusting the code to run according to the slight hardware and software variations on each computer requires finalization work.

5. Schedule


TASKS
11/3
11/15
11/22
11/29
12/6
12/13
12/22
proposal
Function 1
Detailed Function (Function 1)
Function2
Function3
Detailed Function (Function 3)
finishing touches