Python

Developing KakaoTalk automation with Python

Final Report

Date : 2023/12/24

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

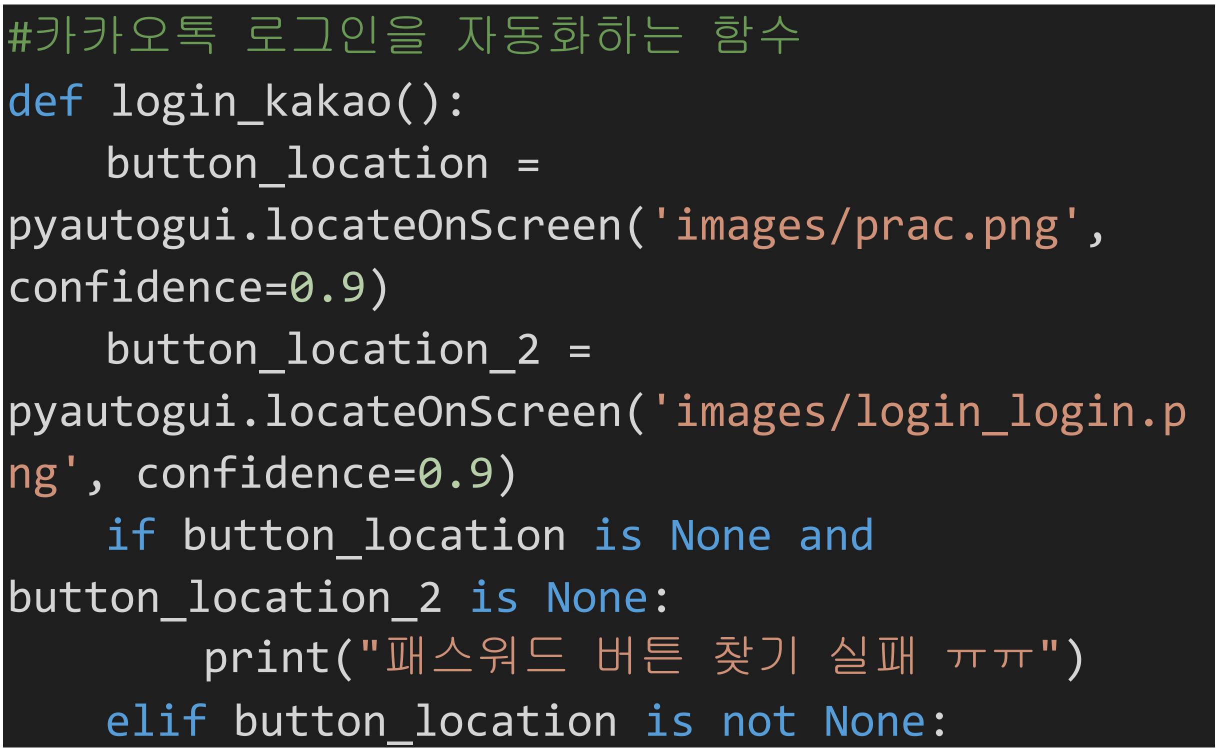
1) Function implementation

1) Function 1

- Add a function to login automatically

(1) Detailed function

- Add a function to close and launch KakaoTalk for initialization



```
#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.png', 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.doubleClick(button_point.x,
button_point.y-45)
        pyautogui.write(PASSWORD)
        pyautogui.press('enter')
```

2) Function 2

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

```
#친구 찾기
def find_fren(fren):
    button_location = None
    button_location =
pyautogui.locateOnScreen('images/search_icon.p
ng', 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)




        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_point.x, demo_chat_point.y+30)
```

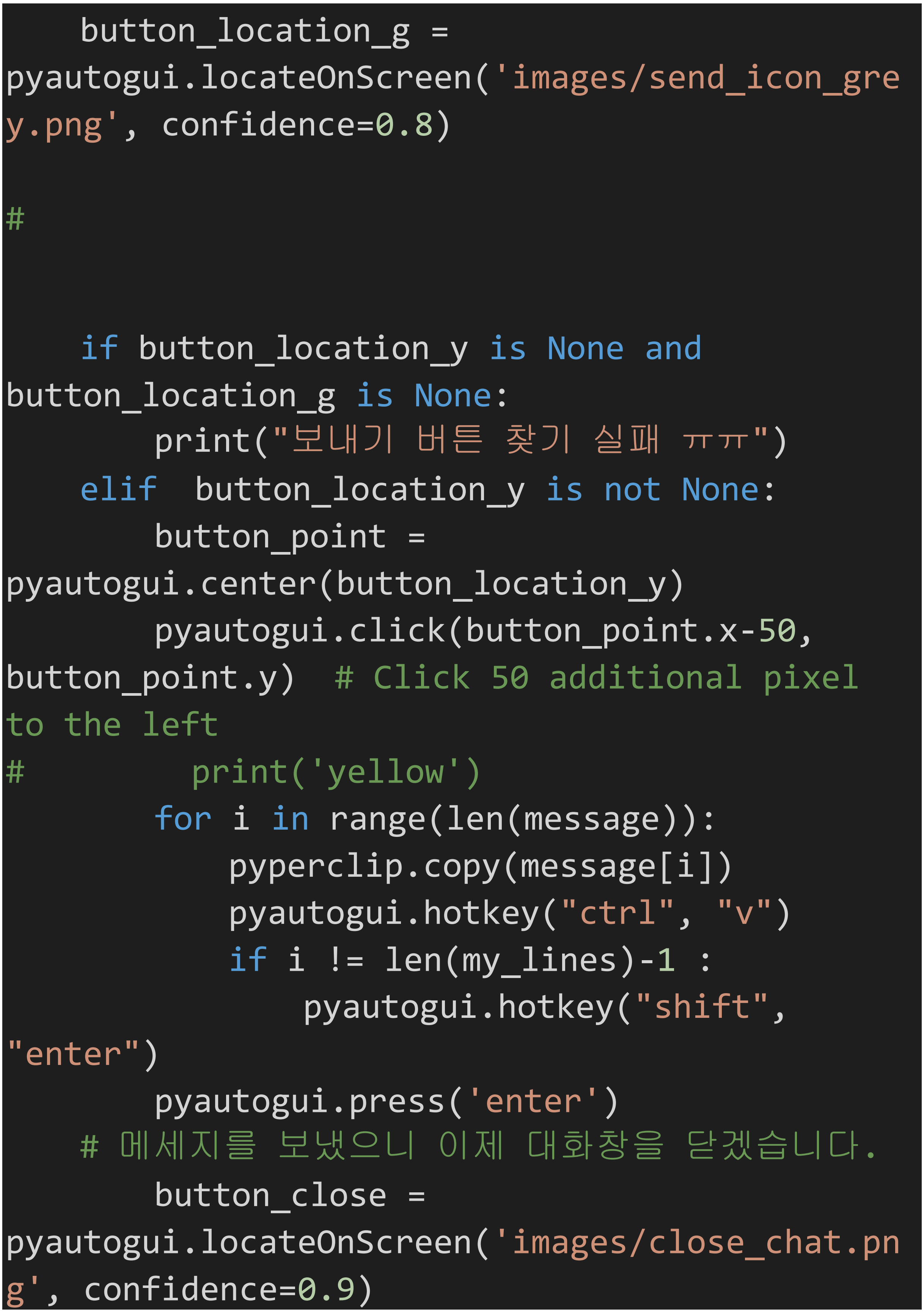
3) Function 3

(1) Detailed function

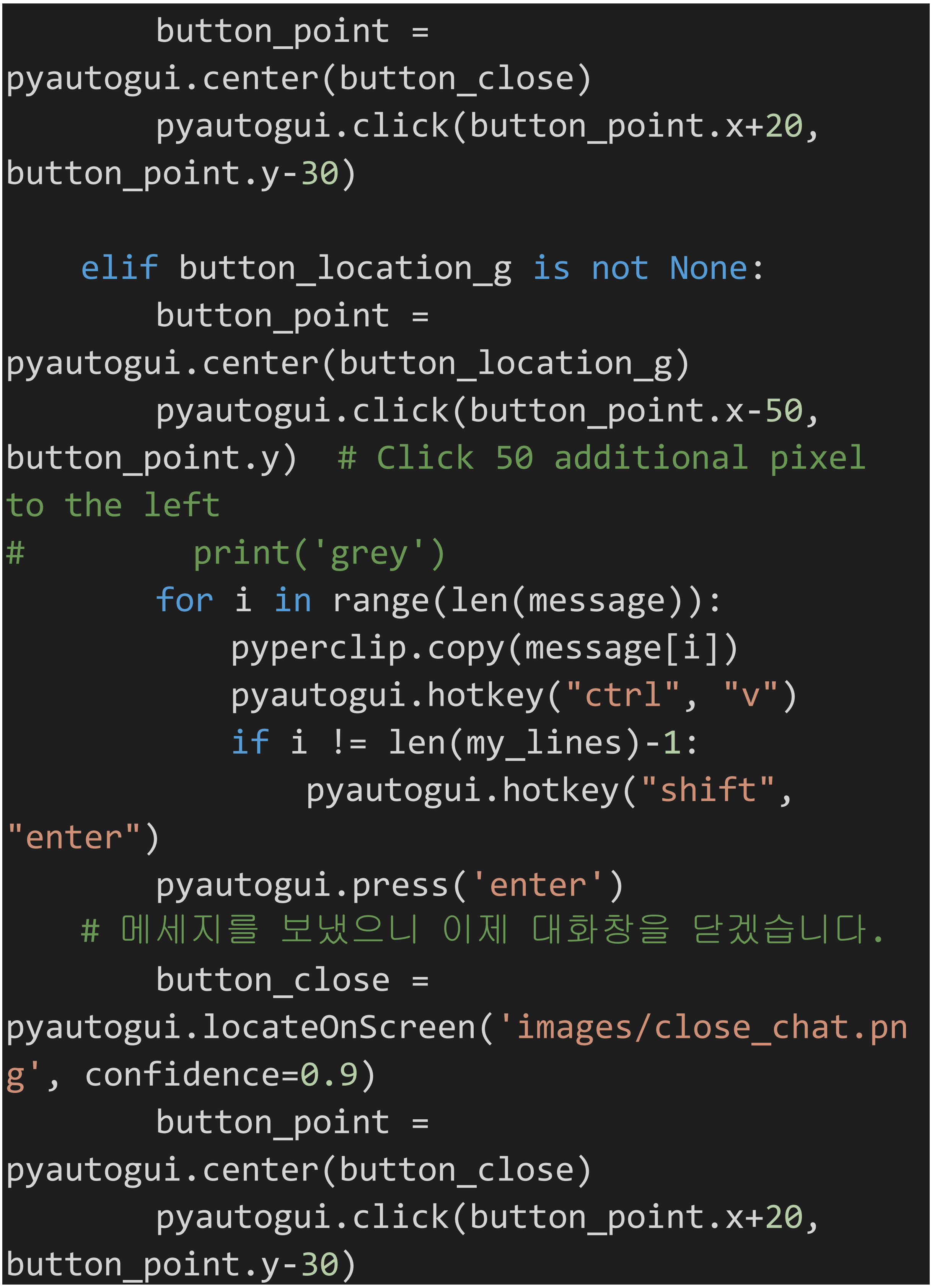
- Add a function to close the chat window after sending a message

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

```
#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)
```

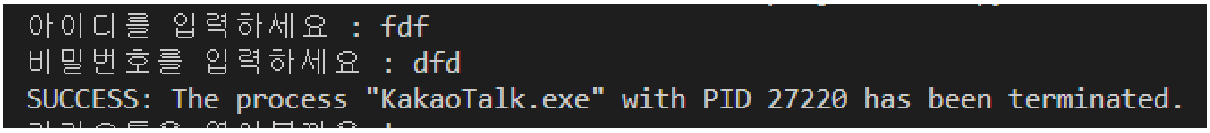


```
        button_point =
pyautogui.center(button_close)
        pyautogui.click(button_point.x+20,
button_point.y-30)

    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.pn
g', confidence=0.9)
        button_point =
pyautogui.center(button_close)
        pyautogui.click(button_point.x+20,
button_point.y-30)
```

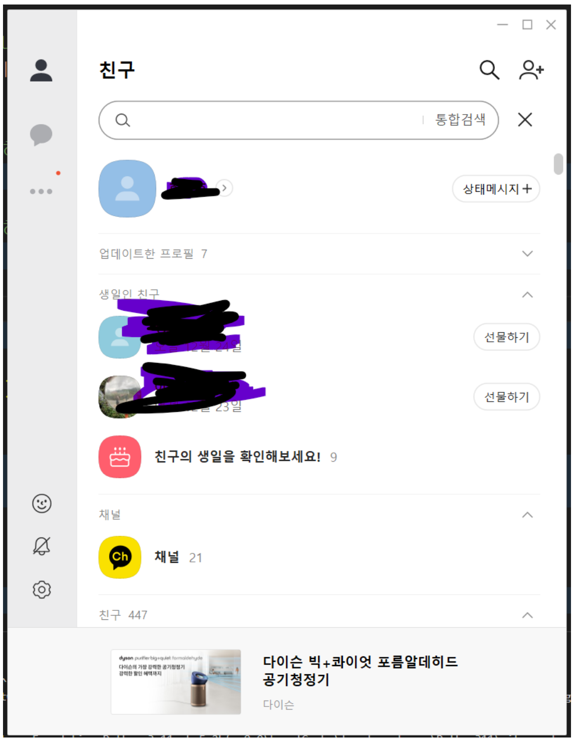
I have done my best to ensure that the code works properly while completing the current project. However, it may work differently depending on the environment. Factors such as the KakaoTalk version, monitor quality, and computer CPU speed may also affect it.

2) Test results

-



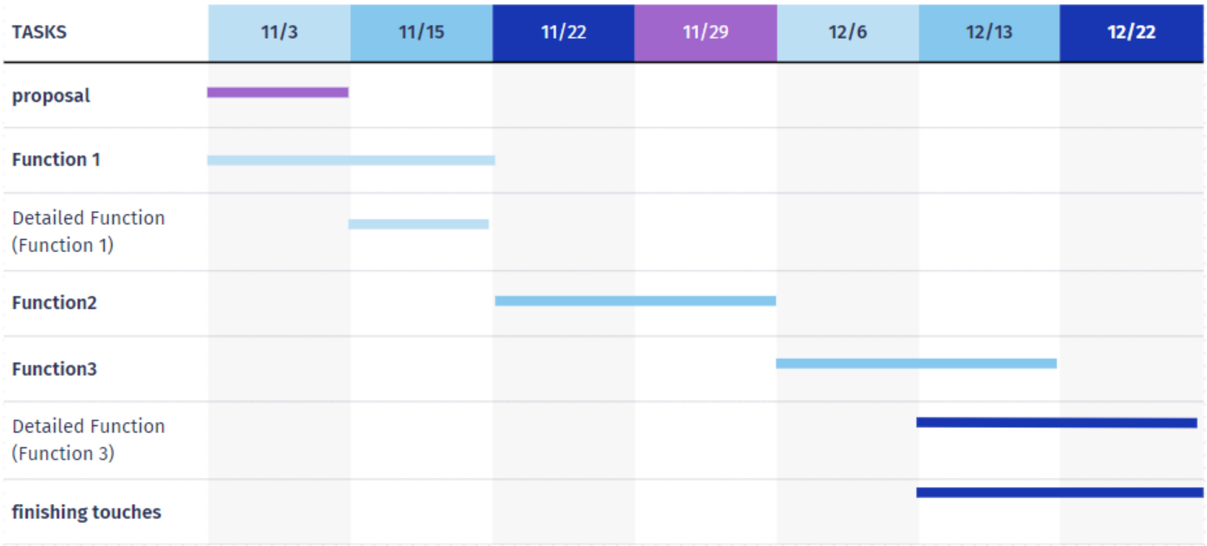
```
아이디를 입력하세요 : fdf
비밀번호를 입력하세요 : dfd
SUCCESS: The process "KakaoTalk.exe" with PID 27220 has been terminated.
```



4. Changes in Comparison to the Plan

- nothing

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						■	■