객체지향 프로그래밍 미니프로젝트

SASA Random Chat

2105 박상권

1. 개발 목적

Motivation

A. 랜덤채팅 어플:

1. **모르는 사람**과 수다를 떨 수 있음 + 모르는 사람이니까 속마음 이야기도 OK
2. 그러다가 정말 **친해지기도 함!**
3. 마니또랑 조금 어딘가 비슷한 구석이 있음!

그렇지만…

1. **위험함**. ‘랜덤’을 뽑는 모집단이 너무 큼
2. 굳이 모르는사람 말고 **내 주변사람에게 익명으로** 말하고 싶을때(대숲처럼 **1:다수가 아닌 1:1**로)가 있음
3. 일단 정말 **위험함**(진짜 랜덤채팅은 저도 해본적 없어요 ><)

1. 개발 목적
Motivation

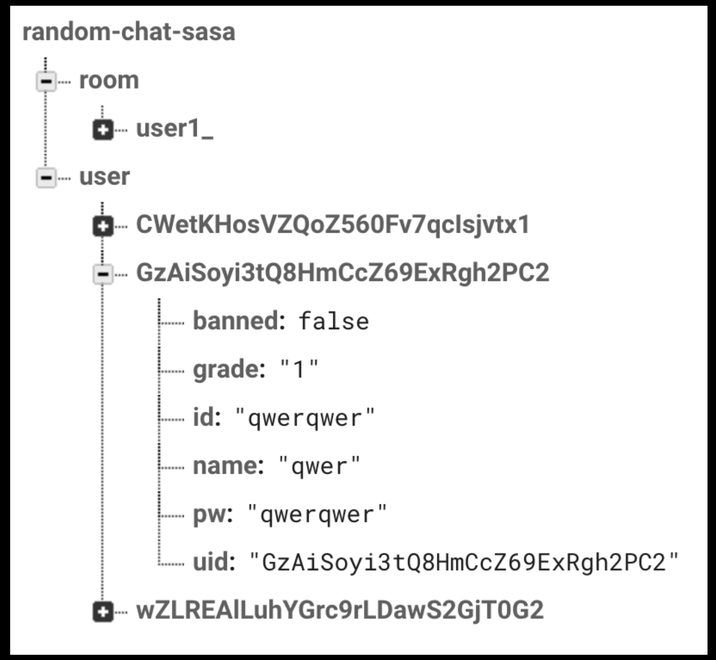
A. 그렇게 등장한…

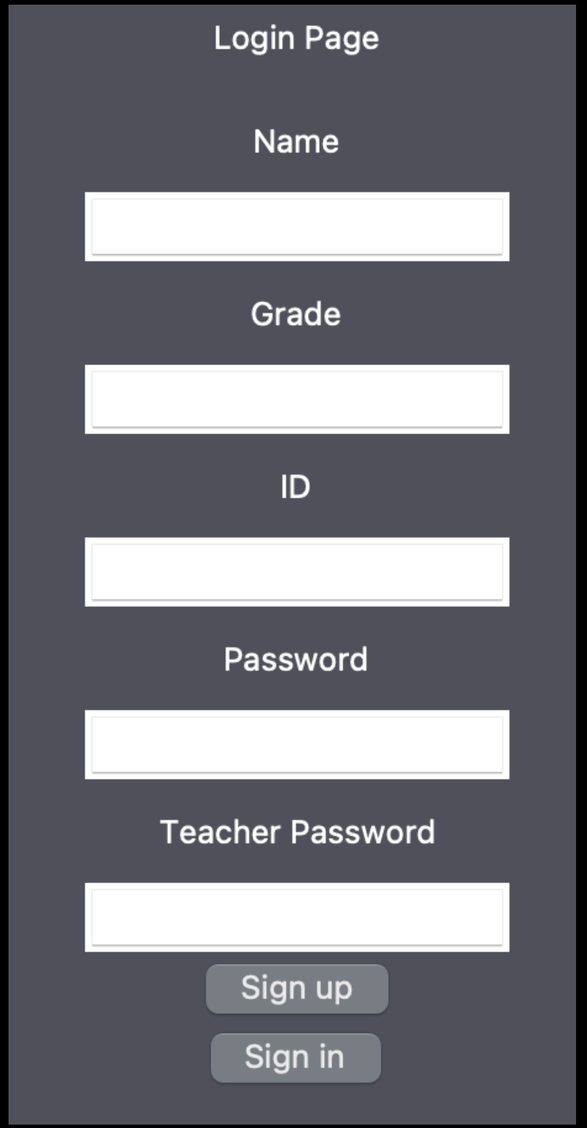
SASA Random Chat

1. ‘랜덤’의 모집단이 **우리학교 == 비교적 매우 안전**
2. 평소 **어색했던 친구들과 가까워**지는 계기가 될 수 있음!
3. 처음 들어와 **물어보고 싶은것도 많고**, 학교에 있는 **모두와 친해지고 싶은 신입생**들에게 매우 효과적!
~~4. 화석들의 놀이터~~

2. 주요기능
Major Functions

A. 로그인 / 유저관리 시스템





2. 주요기능
Major Functions

A. 로그인 / 유저관리 시스템

```
def signup_action(self, controller):# 가입 Method
    tpw = self.tpw.get()
    if tpw == "SASA_":# 관리자 유효 검사
        account = teacher(self.name.get())
    else:
        account = student(self.name.get(), self.grade.get())
    if not account.set_id(self.id.get()):# 아이디 유효성 검사
        messagebox.showerror("Error", "ID is must be longer than 8")
    if not account.set_passwd(self.pw.get()):# 비밀번호 유효성 검사
        messagebox.showerror("Error", "PassWord must be longer than 8")
    flag = account.sign_up()
    if flag:# 가입 성공 여부
        print("going to listpage")
        controller.ListPage_frame_init(account)
        controller.show_frame(ListPage)
    else:# 예외처리
        messagebox.showerror("Error", "Cannot Signup! Report this issue to Developer to fix it.")
```

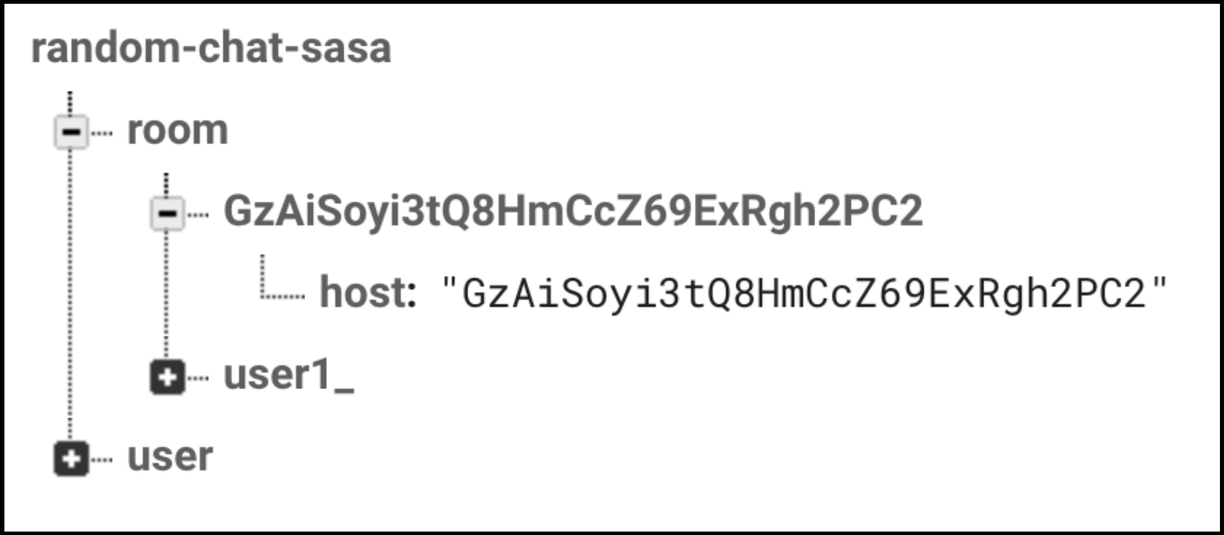
2. 주요기능
Major Functions

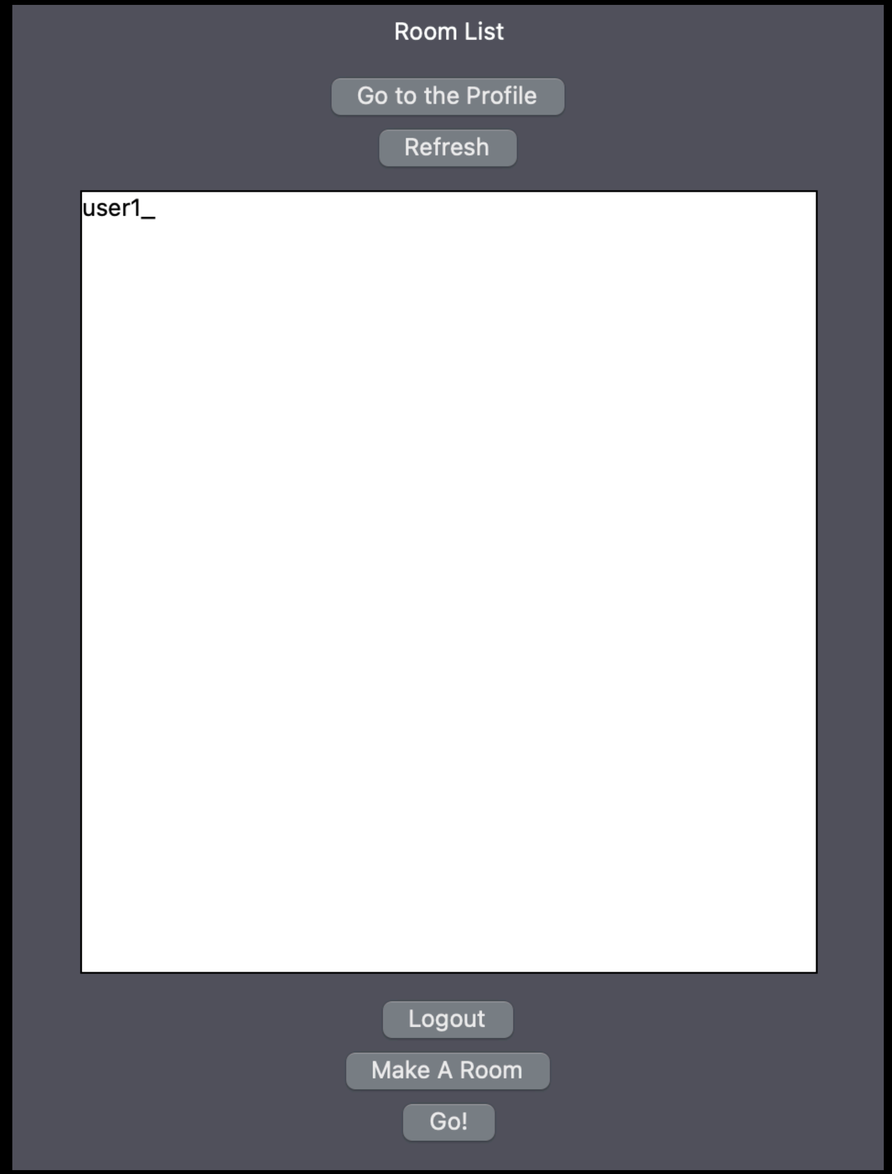
A. 로그인 / 유저관리 시스템

```
def sign_up(self): # person의 기초적 signup 메서드 (유저 생성 및 데이터베이스 저장)
    try:
        user = auth.create_user(
            email=self.__id,
            email_verified=False,
            password=self.__passwd,
            display_name=self.__name,
            disabled=False)
        self.__user = user
        DB = firebase.database()
        DB.child("user").child(user.uid).set({"uid": self.get_user().uid,
                                              "id": self.get_id(),
                                              "pw": self.get_passwd(),
                                              "name": self.get_name()})
        print('Sucessfully created new user: {0}'.format(user.uid))
        return True
    except Exception as e: # 예외처리
        print(e)
        print("This ID was Already used. Try Again with another ID")
        return False
```

2. 주요기능
Major Functions

A. 대기실 생성 및 입장





2. 주요기능
Major Functions

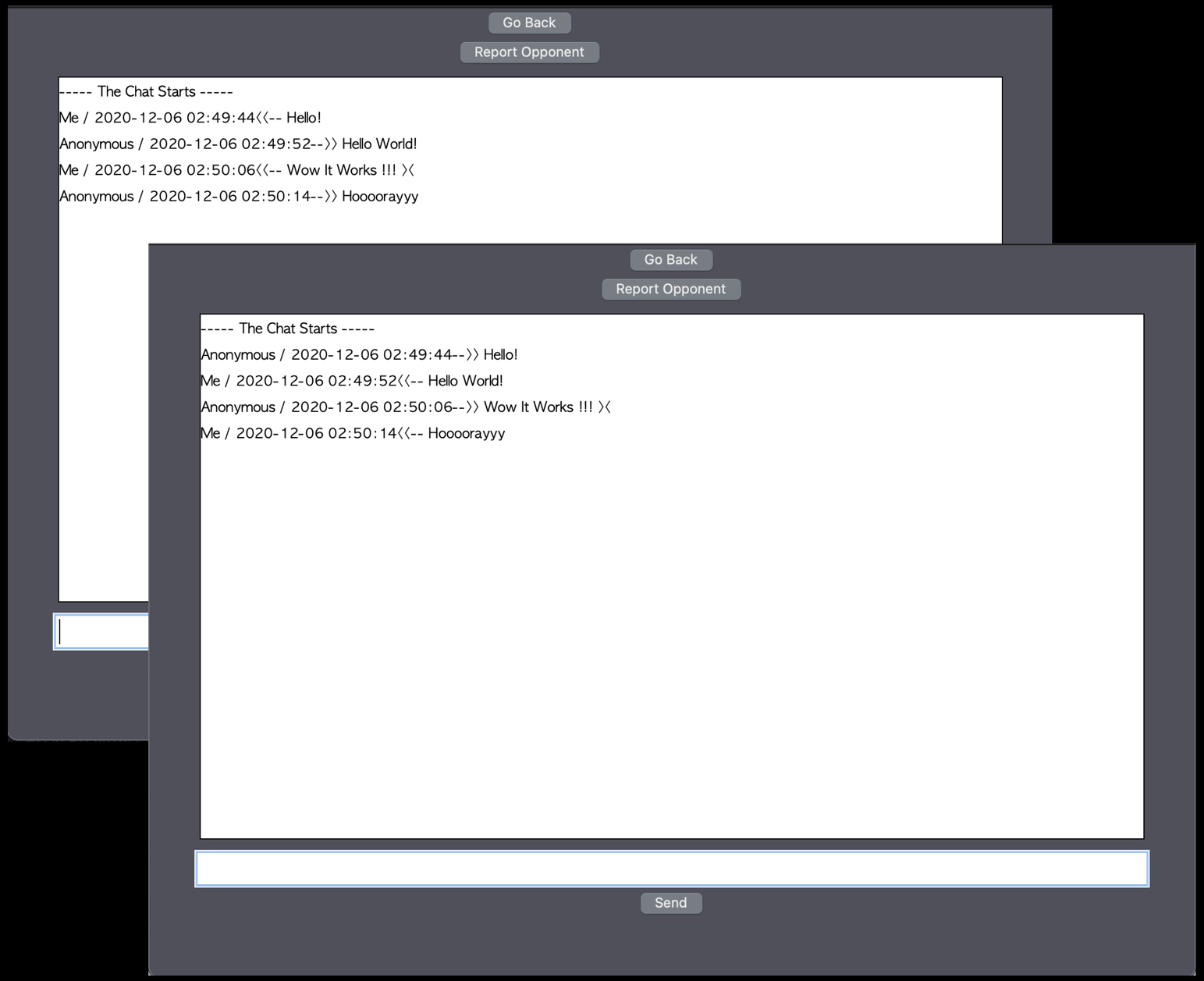
A. 대기실 생성 및 입장

```
def wait_action(self, controller, account): # 대기 페이지 이동 메서드
    print("waiting")
    print(account.get_user().uid)
    try:
        DB = firebase.database()
        DB.child("room").child(account.get_user().uid).set({"host": account.get_user().uid}) # 데이터베이스에 방 생성
        controller.WaitPage_frame_init(account)
        controller.show_frame(WaitPage)
    except: # 예외 처리
        print("Set Failed")
```

```
def go_action(self, controller, account): # 방 참가 + 창 이동 메서드
    host = self.listNodes.get(self.listNodes.curselection()[0])
    db.child("chat").child(host + "+" + account.get_user().uid).set({"chat_count": 1, "exist_": 2}) # 채팅 기본값 데이터베이스에 저장
    db.child("chat").child(host + "+" + account.get_user().uid).child("messages").child(0).set({"number": 0,
                                                                                               "data": {
                                                                                                   "date": datetime.datetime.now().strftime(
                                                                                                       '%Y-%m-%d %H:%M:%S'),
                                                                                                   "from": "public",
                                                                                                   "message": "----- The Chat Starts -----"}}) # 첫 공식 채팅 데이터베이스에 저장

    db.child("room").child(host).update({"host": account.get_user().uid})
    controller.ChatPage_frame_init(account, host + "+" + account.get_user().uid)
    controller.show_frame(ChatPage)
```

2. 주요기능
Major Functions

A. 실시간 채팅



2. 주요기능
Major Functions

A. 실시간 채팅

```
def stream_handler(self, message, account):  # 스트림 핸들러 - 이벤트 발생시 호출
    print("handled")
    print(message)
    try:
        print(message['data'])
        message = message['data']
        print(message['data'])
        message = message['data']
        print(message)
        if message['from'] == account.get_user().uid:# 내가 보낸 채팅
            self.listNodes.insert(self.listNodes.size(),
                                  "Me" + " / " + message['date'] + "<<-- " + message['message'])
            print(message['from'] + " / " + message['date'] + "<<-- " + message['message'])
        elif message['from'] == "public":# 공식 채팅
            self.listNodes.insert(self.listNodes.size(), message['message'])
            print(message['message'])
        else:# 상대가 보낸 채팅
            self.listNodes.insert(self.listNodes.size(),
                                  "Anonymous" + " / " + message['date'] + "-->> " + message['message'])
            print(message['from'] + " / " + message['date'] + "-->> " + message['message'])
    except Exception as e:# 예외처리
        print("Not Yet")
        print(e)
```

2. 주요기능
Major Functions

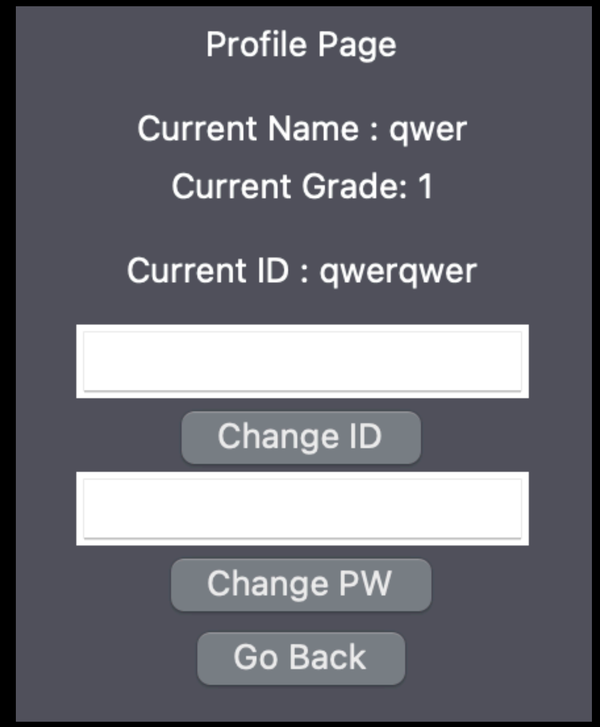
A. 실시간 채팅

```
def send_chat(self, account, roomname): # 채팅 전송 메서드
    cnt = db.child("chat").child(roomname).get().val()['chat_count']
    m = unicodedata.normalize('NFC', self.chat.get())
    db.child("chat").child(roomname).child("messages").child(cnt).set({"number": cnt,
                                                                       "data": {
                                                                           "date": datetime.datetime.now().strftime(
                                                                               '%Y-%m-%d %H:%M:%S'),
                                                                           "from": account.get_user().uid,
                                                                           "message": m}}) # 대화 객체 (Dictionary Type) 데이터베이스에 저장
    cnt += 1
    db.child("chat").child(roomname).update({"chat_count": cnt}) # 대화 카운트 증가
    self.chat.delete(0, "end") # 입력칸 비우기
```

2. 주요기능
Major Functions

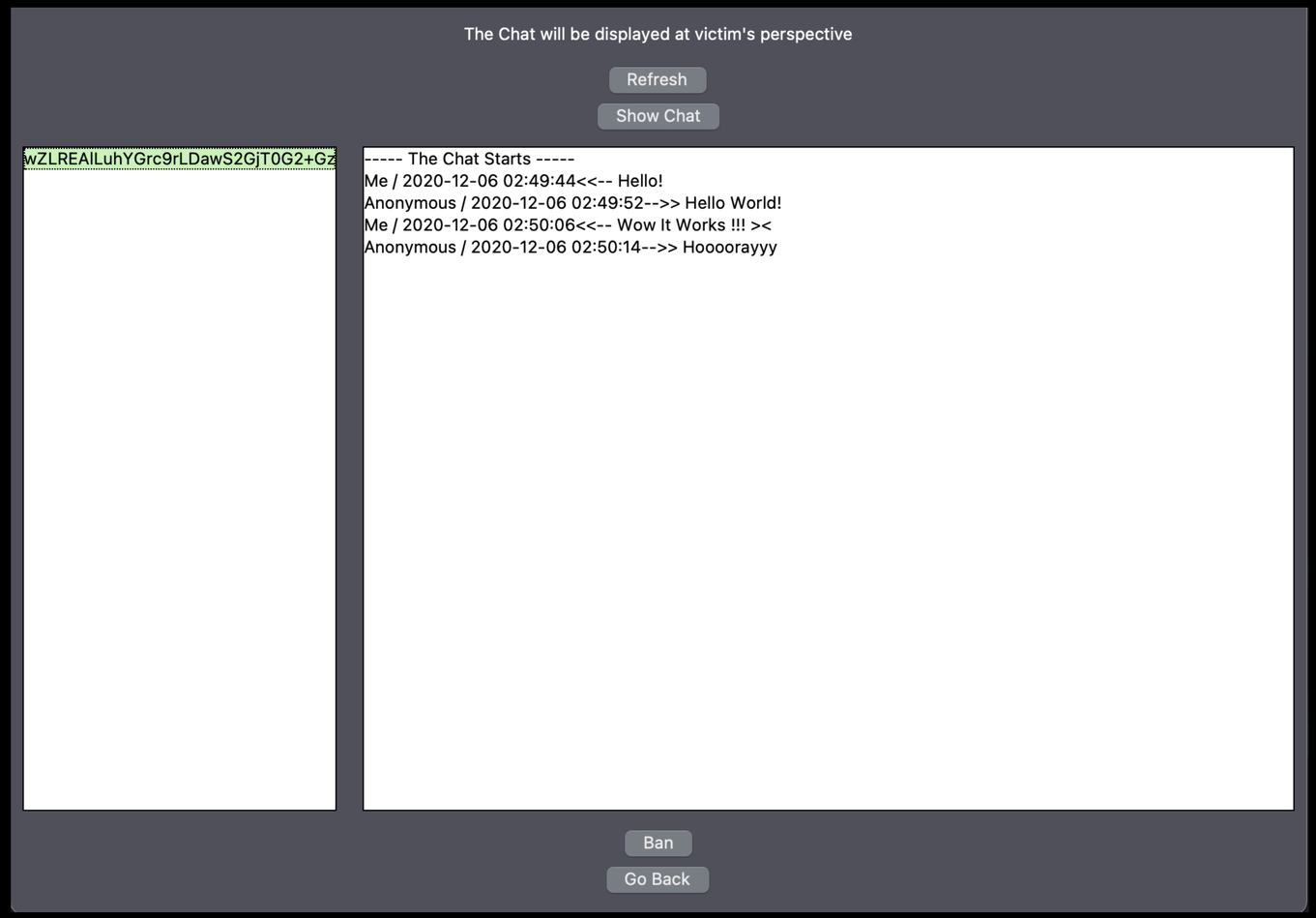
A. 프로필 관리

```
def go_back_action(self, controller, account):# 뒤로가기 메서드
    controller.ListPage_frame_init(account)
    controller.show_frame(ListPage)

def id_cnage(self, account):# 아이디 바꾸기 메서드
    if account.set_id(self.id.get()):
        account.update_value()
        messagebox.showinfo("Info", "Successfully Changed ID")
    else:# 예외처리
        messagebox.showerror("Error", "ID must be longer than 8")

def pw_change(self, account):# 비밀번호 바꾸기 메서드
    if account.set_passwd(self.pw.get()):
        account.update_value()
        messagebox.showinfo("Info", "Successfully Changed Passwd")
    else:# 예외처리
        messagebox.showerror("Error", "PW must be longer than 8")
```



2. 주요기능
Major Functions

A. (관리자) 신고 관리 및 사용자 정지



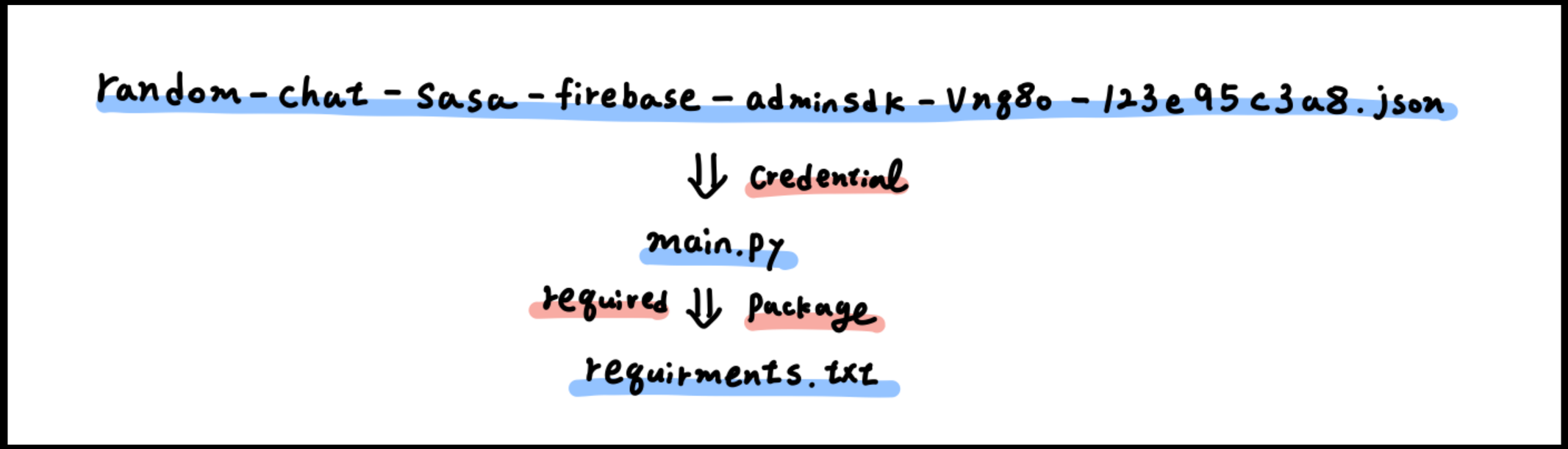
2. 주요기능
Major Functions

A. (관리자) 신고 관리 및 사용자 정지

```
def ban_action(self, account):# 해당 사용자 차단 메서드
    roomname = self.listNodes.get(self.listNodes.curselection()[0])# 사건 일어난 방 번호
    target = db.child("request").child(roomname).get().val()['target']# 피의자 UID
    print(target)
    if not account.ban(target):# 예외처리 (차단 불가능)
        messagebox.showerror("Error", "Unexpected Error : Cannot Ban the target")
    else:
        db.child("request").child(roomname).remove()# 신고내역 지우기
        db.child("ban_req").child(roomname).remove()
        self.refresh_list()
        self.Chat.delete(0, tk.END)
        messagebox.showinfo("Info", "Successfully Banned \n " + target)
```

```
def report_action(self, account, roomname):# 신고 메서드
    DB = firebase.database()
    l = roomname.split("+")
    target = ""
    for i in l:
        if i != account.get_user().uid:
            target = i
    DB.child("request").child(roomname).set({"victim": account.get_user().uid,
                                             "target": target})# request에는 신고 대상자와 신고자 uid 목록이 저장됨
    context = DB.child("chat").child(roomname).child("messages").get()# 채팅 데이터 가져오기
    DB.child("ban_req").child(roomname).set(context.val())# 신고목록에 채팅데이터 저장
```

3. 파일 구성도
File Configuration



4. 클래스 관계도
Class Configuration