

---

# 카카오톡 대화방 관리 구현(E조)

---

C팀 : 박정호, 김진호, 신재하, 김수진

---

01. 서론

---

02. 구현

---

03. 구현 문제점

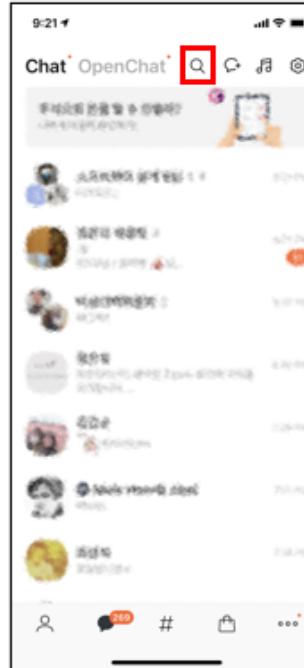
---

04. QnA

---

## 통합검색

1. 사용자는 검색어를 입력하여 원하는 대화방 또는 친구 목록을 확인, 입장할 수 있다.
2. 시스템은 사용자가 입력된 검색어가 포함된 목록을 출력한다.
3. 사용자가 원한다면 최근 검색어를 자동 저장하고 삭제 할 수 있다.



[A] 통합 검색 버튼



[B] 검색어 입력칸



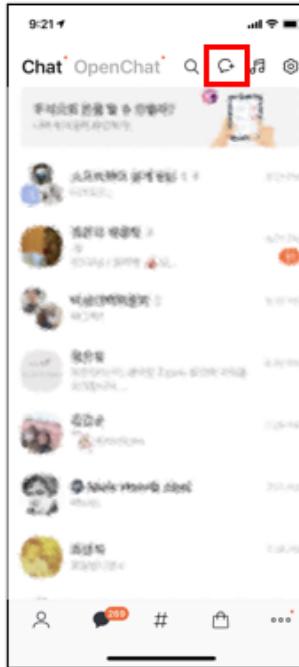
[C] 검색어 입력



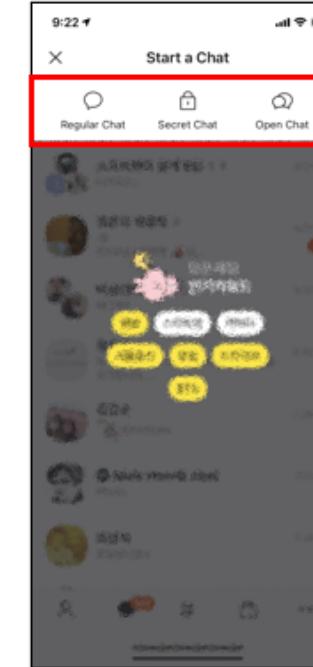
[D] 최근 검색어 자동 저장 및 삭제

## 대화방 생성

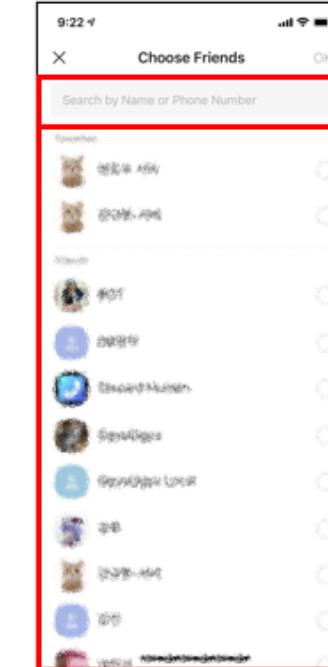
1. 사용자가 초대 및 참가 기능으로 함께 채팅 할 수 있는 **대화방을 생성할 수 있다.**
2. 사용자가 대화방 생성 시 **대화방의 유형을 선택할 수 있다.**
3. 사용자가 **대화방에 초대할 대상을 선택할 수 있다.**



[A] 대화방 생성 버튼



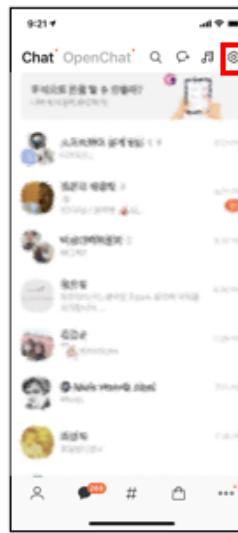
[B] 대화방 유형 선택



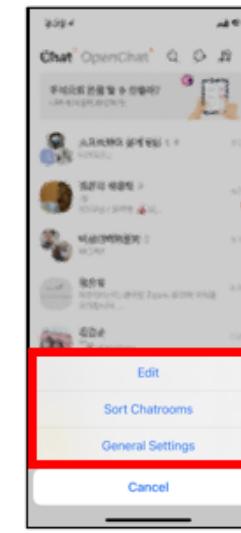
[C] 초대 대상 선택

# 대화방 설정

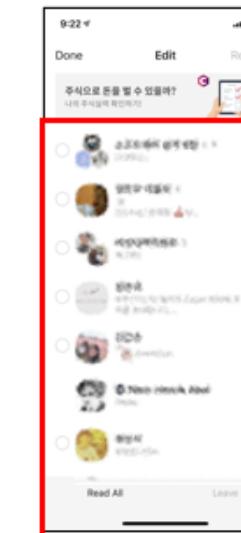
1. 대화방 설정을 관리할 수 있는 기능으로 편집, 정렬을 포함한다.
2. 여러 개의 대화방을 한 번에 읽음 처리하거나 나갈 수 있다.
3. 대화방을 원하는 순서로 정렬할 수 있다.
4. 대화방 목록을 그룹화 할 수 있다.
5. 대화방을 병합 할 수 있다.



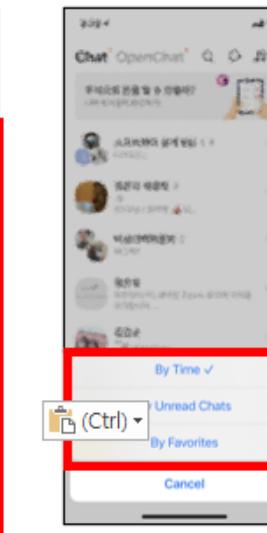
[A] 설정 버튼



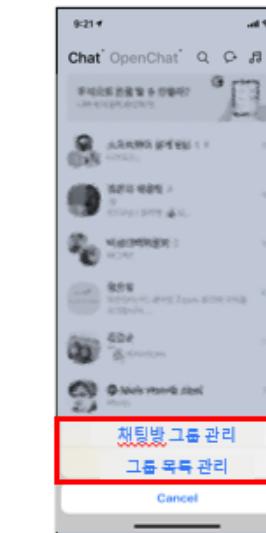
[B] 설정 선택



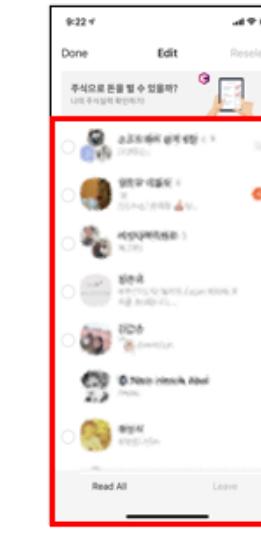
[C] 편집



[D] 정렬



[E] 그룹화

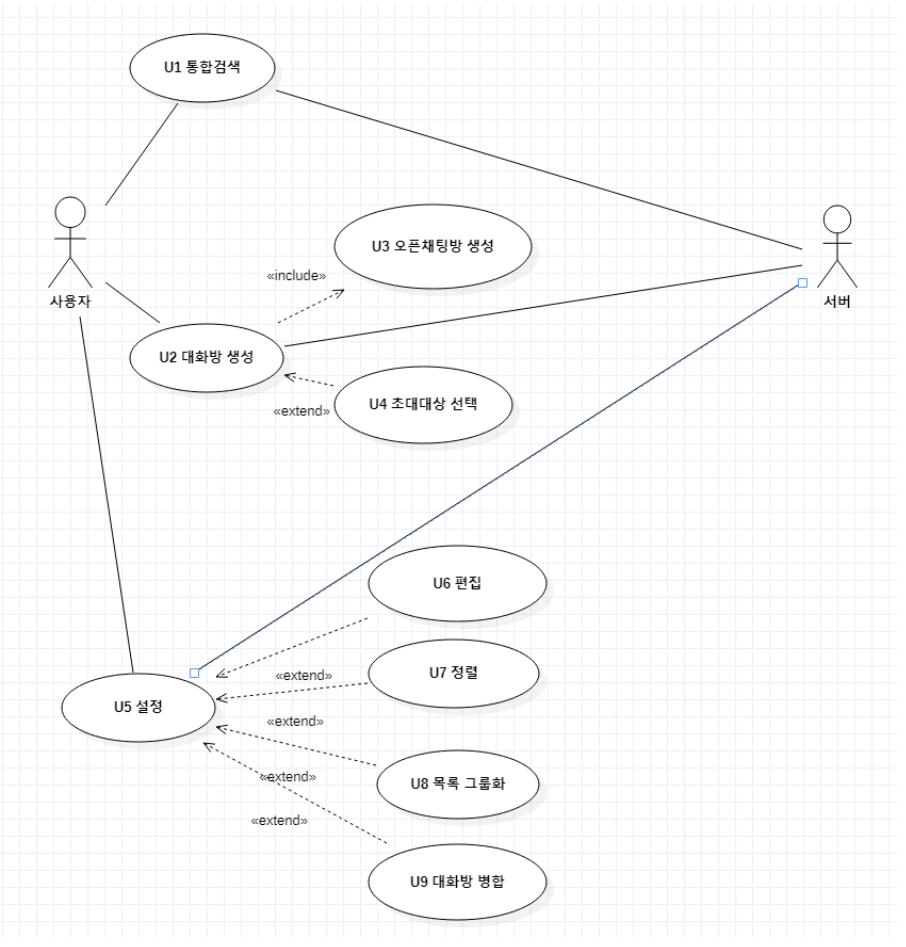


[F] 병합

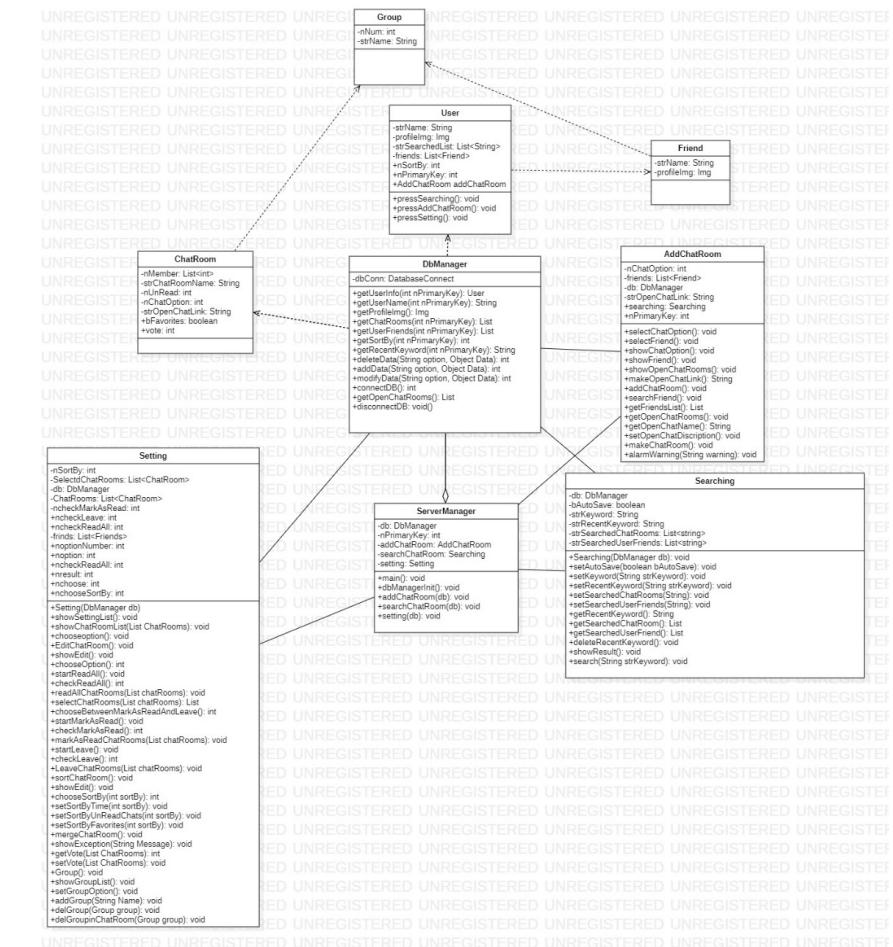
H/W	Cpu	Intel® Core™ i7 – 7700HQ
	Ram	8192MB
	Storage	SSD 237GB
S/W	OS	Windows 10 64bit
	Language	Java
	Tool	Eclipse

표 : 개발 환경

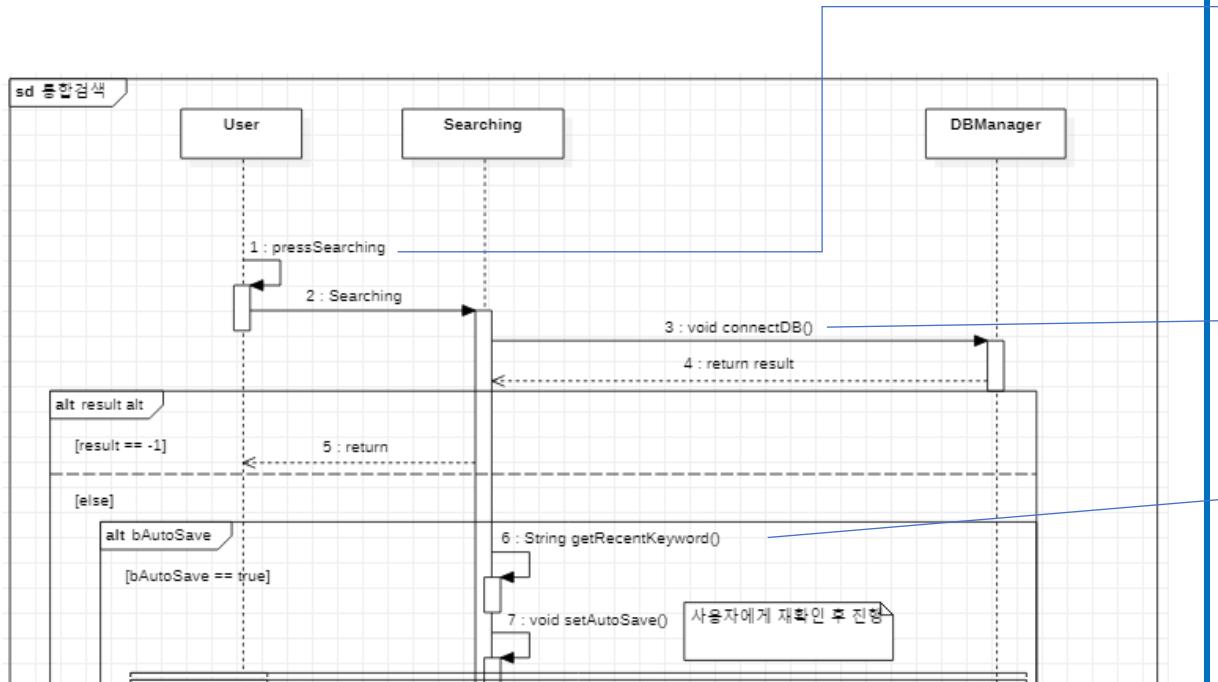
## 유스케이스 다이어그램



## 클래스 다이어그램



## 시퀀스 다이어그램



## 구현 코드

```

private static Searching searchChatRoom = new Searching();

public static void main(String[] args) {
    user.pressSearching(); // 사용자가 검색버튼을 클릭
    searchChatRoom.Searching(db, nPrimaryKey); // 검색 기능 시작
}

public void Searching(DbManager database, int PrimaryKey) {
    db = database;
    nPrimaryKey = PrimaryKey;
    String searchKeyword;
    int result;
    if(db.connectDB() == -1) { // db연결에 실패할 경우
        System.out.println("DB연결 실패!!!");
        return;
    }
    else { // db연결에 성공
        if(bAutoSave == true) { // 자동저장기능이 켜져있을 경우
            strRecentKeyword.add(getRecentKeyword()); // DB에서 최근 검색어 가져옴
            setAutoSave(); // 자동저장기능을 유지할 것인지 사용자에게 재확인
        }
    }
}

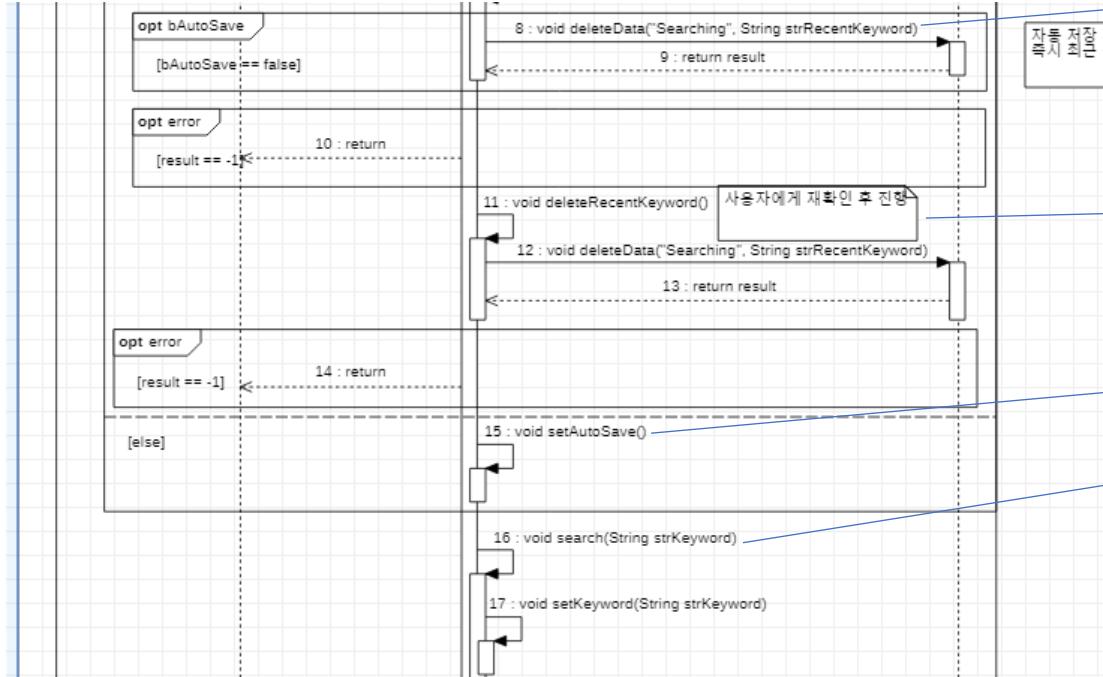
public String getRecentKeyword() {
    String recentKeyword = db.getRecentKeyword(nPrimaryKey); // DB에서 검색어 가져옴
    return recentKeyword;
}

public void setAutoSave() {
    Scanner sc = new Scanner(System.in);
    System.out.println("setAutoSave (1. true, 2. false)");
    int i = sc.nextInt();
    if(i == 1) {
        this.bAutoSave = true;
    }
    else if(i == 2) {
        System.out.println("Stop AutoSave? \"y\", \"n\"");
        if(sc.next() == "y") {
            this.bAutoSave = false;
        }
        else {
            return;
        }
    }
}

```

The implementation code follows the sequence diagram. It initializes the Searching object and calls its constructor with the database and primary key. In the constructor, it attempts to connect to the database. If successful, it checks if auto-save is enabled. If so, it adds the recent keyword to a list and calls the setAutoSave() method to prompt the user for confirmation. The getRecentKeyword() method retrieves the recent keyword from the database. The setAutoSave() method uses a Scanner to get user input (1 for true, 2 for false) and updates the bAutoSave boolean accordingly.

## 시퀀스 다이어그램



## 구현 코드

```

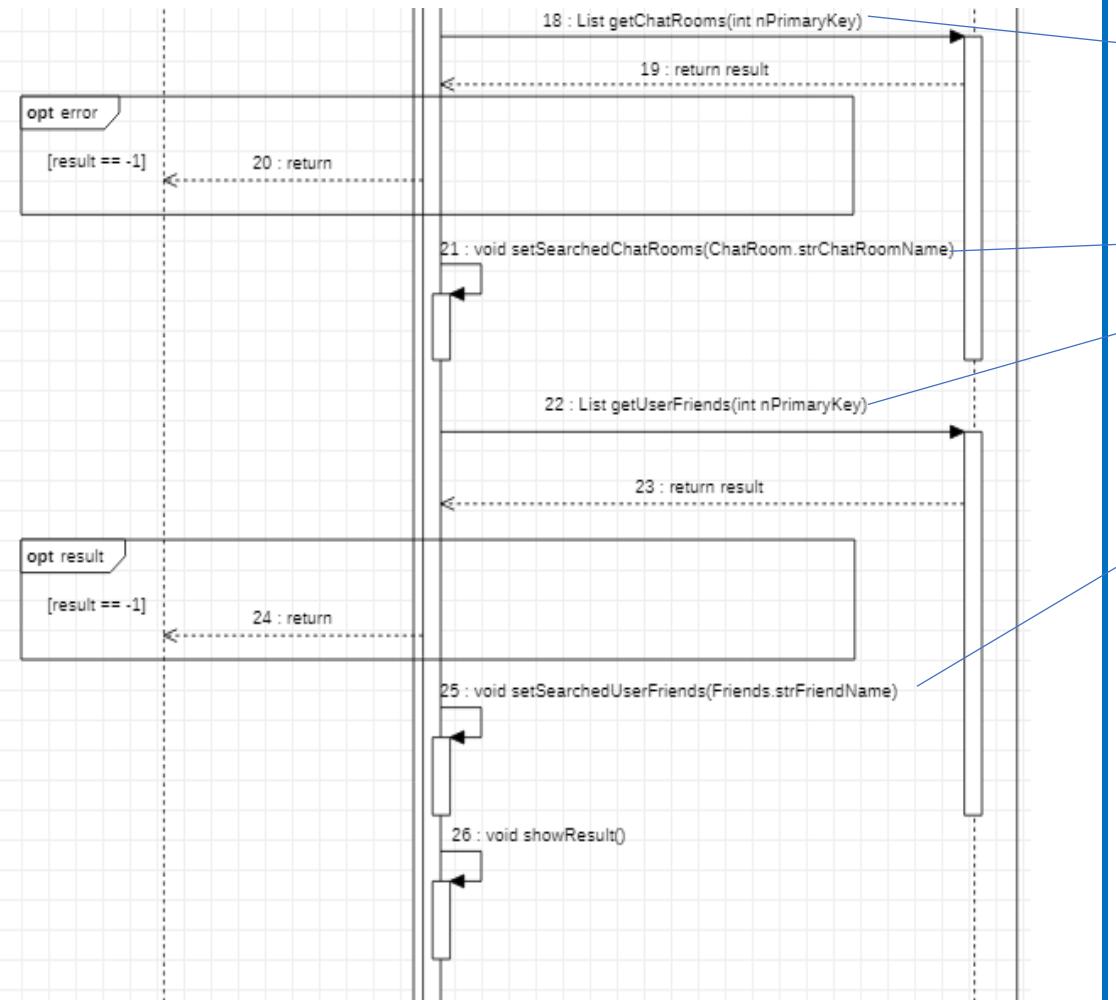
public void Searching(DbManager database, int PrimaryKey) {
    if(bAutoSave== false) { //자동저장기능을 끌 경우
        result = db.deleteData("Searching",getRecentKeyword()); //db의 최근검색 기록 삭제
        if(result == -1) { //db의 비정상적인 종료 등의 오류
            return;
        }
    }
    System.out.println("최근 검색어를 삭제하시겠습니까?");
    /*if(deleteRecentKeyword()) { //사용자가 최근 검색어 삭제를 요청한 경우
        result = db.deleteData("Searching",getRecentKeyword()); //db의 최근검색 기록 삭제
        if(result == -1) { //db의 비정상적인 종료 등의 오류
            return;
        }
    }*/
    else { //자동저장기능이 꺼져있을 경우
        setAutoSave(); //자동저장 기능을 사용할 것인지 사용자에게 확인
    }
    System.out.print("검색어를 입력해 주세요. \n> ");
    search(sc.next()); //사용자의 입력을 통해 검색
}

public void search(String strKeyword) {
    List list;
    setKeyword(strKeyword);
}

public void setKeyword(String strKeyword) {
    this.strKeyword=strKeyword;
    System.out.println("키워드설정");
}
...

```

## 시퀀스 다이어그램



## 구현 코드

```

public void Searching(DbManager database, int PrimaryKey) {
    list = db.getChatRooms(nPrimaryKey, strKeyword);
    if(list == null) {
        System.out.println("채팅방 가져오기 실패");
        return;
    }
    setSearchedChatRooms(list);
    list = db.getUserFriends(nPrimaryKey);
    if(list == null) {
        System.out.println("유저 목록 가져오기 실패");
        return;
    }
    setSearchedUserFriends(list);
    showResult();
}

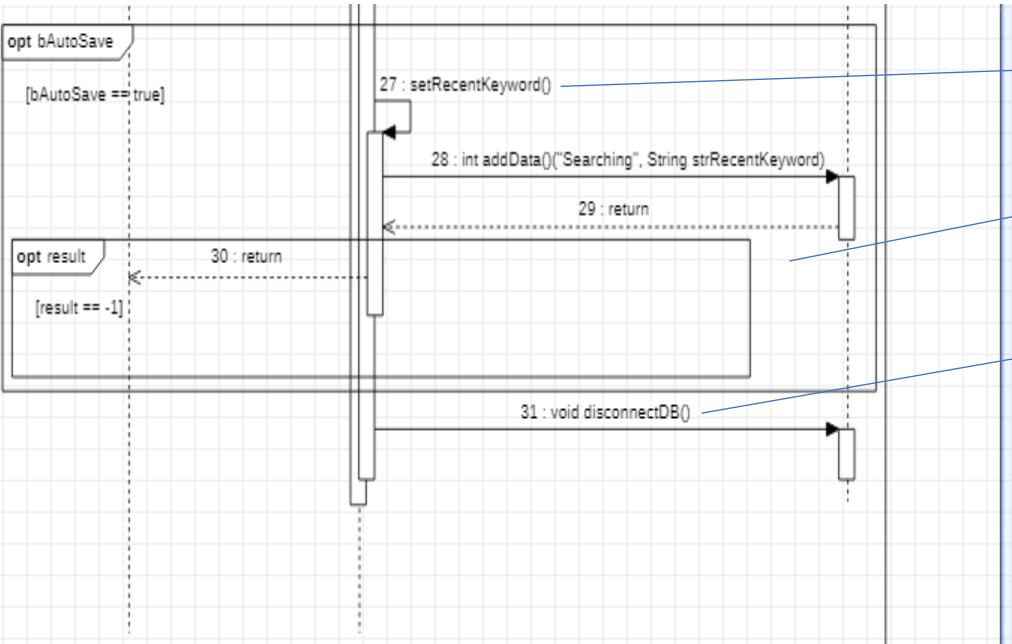
public void setSearchedUserFriends(ArrayList<User> userFriends) {
    this.SearchedUserFriends=userFriends;
}

public void setSearchedChatRooms(ArrayList<ChatRoom> chatRooms) {
    this.SearchedChatRooms=chatRooms;
    System.out.println("채팅방목록설정");
}

public void showResult() {
    System.out.println("검색 결과보여주기");
}
  
```

The implementation code follows the sequence diagram. It first checks if the database query for chat rooms returns null. If so, it prints an error message and returns. Otherwise, it calls the `setSearchedChatRooms` method. It then performs a similar check and call for user friends. Finally, it calls the `showResult` method to print the search results.

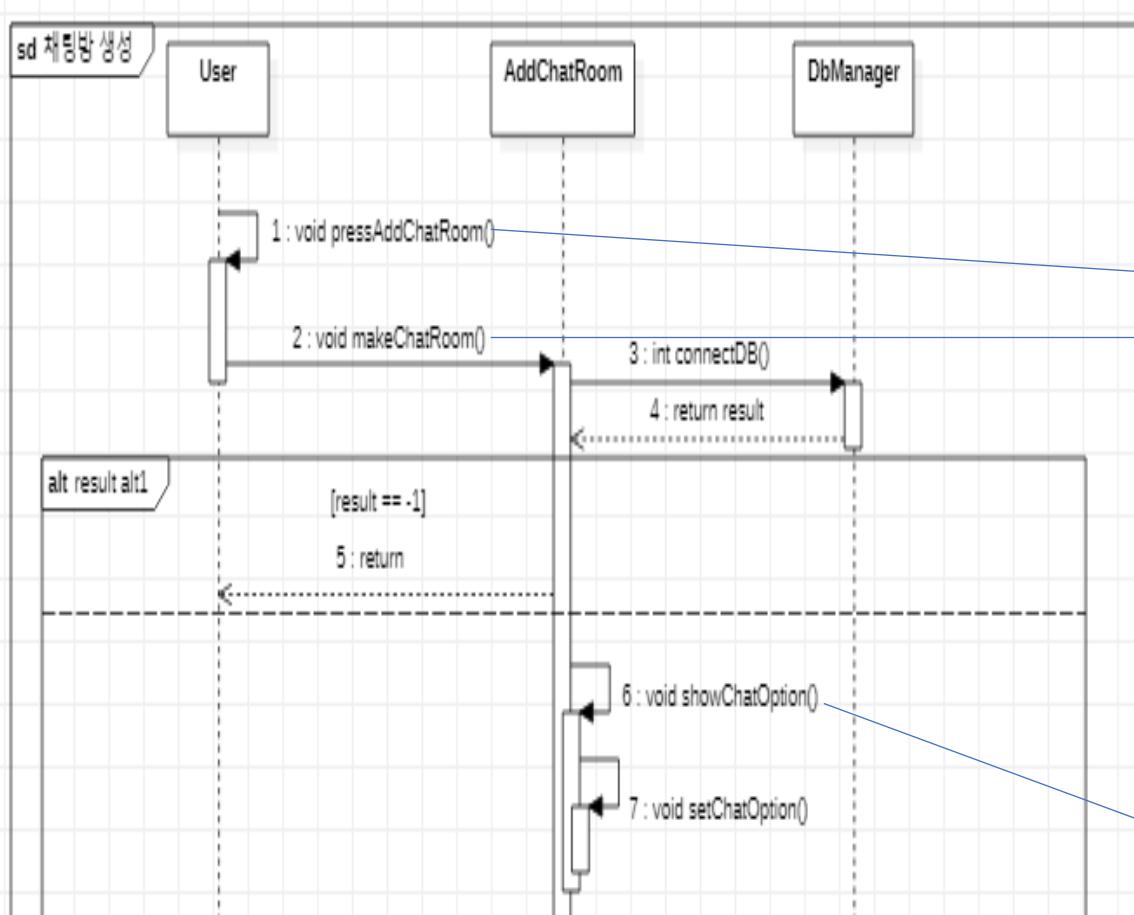
## 시퀀스 다이어그램



## 구현 코드

```
public void Searching(DbManager database, int PrimaryKey) {
    if(getAutoSave() == true) {
        setRecentKeyword(strKeyword);
        result = db.addData("Searching", getRecentKeyword());
        if(result == -1) {
            return;
        }
        db.disconnectDB();
    }
}
```

## 시퀀스 다이어그램



## 구현 코드

```

public class ServerManager {
    private static DbManager db = new DbManager();
    private static int nPrimaryKey;
    private static AddChatRoom addChatRoom = new AddChatRoom();

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        User user = new User();
        user.pressAddChatRoom();
        makeChatRoom(db);
    }

    public static void makeChatRoom(DbManager db) {
        addChatRoom.makeChatRoom(db);
    }

    public void makeChatRoom(DbManager database) {
        this.db=database;

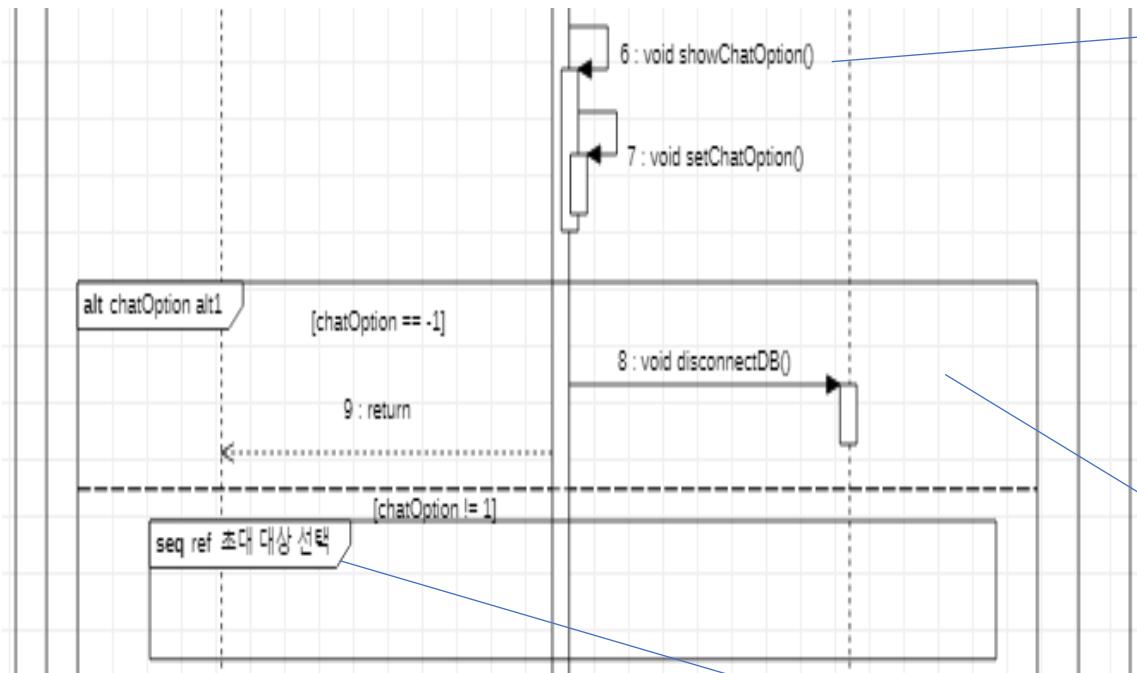
        int result;
        result = db.connectDB();

        if(result == -1) {
            return;
        } else {
            showChatOption();
            if(nChatOption == -1) {
                db.disconnectDB();
                return;
            }
        }
    }
}

```

AddChatRoom 클래스

## 시퀀스 다이어그램



## 구현 코드

```

public void showChatOption() { //채팅방옵션 보여주기
    selectChatOption(); //채팅방옵션 선택
}

```

```

public void selectChatOption() { //채팅방 옵션 선택
    Scanner sc = new Scanner(System.in);
    nChatOption = sc.nextInt();
}

```

```

public void makeChatRoom(DbManager database) {
    this.db=database;
}

```

```

int result;
result = db.connectDB();

```

```

if(result == -1) {
    return;
}

```

```

else {
    showChatOption();
    if(nChatOption == -1) {
        db.disconnectDB();
        return;
    }
}

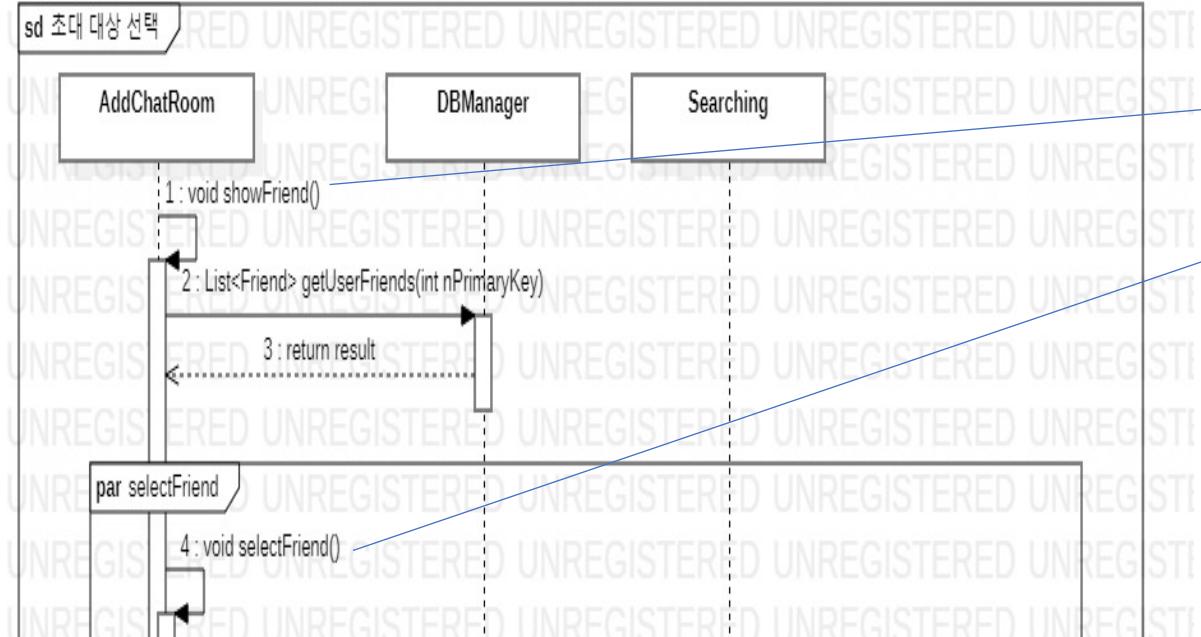
```

```

else if(nChatOption != 1) { //일반채팅
    showFriend(); //초대대상선택레퍼런스
    if(friends.size() > 0) {
        addChatRoom();
        db.disconnectDB();
    }
}

```

## 시퀀스 다이어그램



## 구현 코드

```

public void showFriend() {
    friends = db.getUserFriends(nPrimaryKey);
    selectFriend();
}

DbManager 클래스

public List<Friend> getUserFriends(int nPrimaryKey) {
    List<Friend> list = new ArrayList<Friend>();
    Friend friend = new Friend();
    list.add(friend); //DB에서 가져왔다고 가정
    //사용자에게 친구목록을 보여준다.
    return list;
}

public void selectFriend() { //친구목록에서 초대할 친구 선택
    searchFriend(); //친구 검색
}
  
```

The implementation code for the `showFriend()` method in the `Searching` class is as follows:

- It calls `getUserFriends(nPrimaryKey)` on the `DbManager` object.
- It then calls `selectFriend()`.

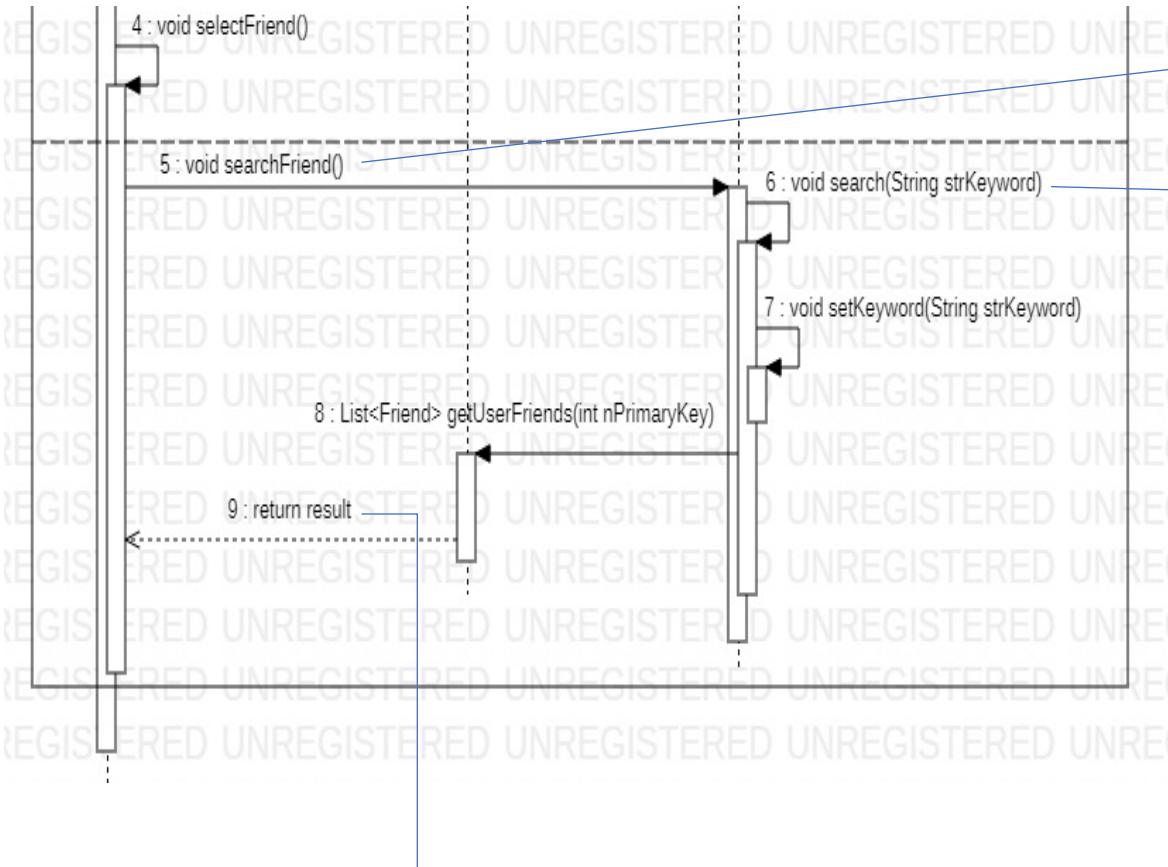
The `getUserFriends` method is implemented in the `DbManager` class:

- It creates a new `List<Friend>` named `list`.
- It creates a new `Friend` object named `friend`.
- It adds `friend` to `list`. A comment indicates this is assumed to be fetched from the DB.
- It returns the `list`.

The `selectFriend()` method is also implemented in the `Searching` class:

- It calls `searchFriend()`.

## 시퀀스 다이어그램



## 구현 코드

```

public void searchFriend() {
    Scanner sc = new Scanner(System.in);
    String str = sc.next(); // 사용자의 입력
    searching.search(str);
}

public void search(String strKeyword) {
    setKeyword(strKeyword);
    db.getUserFriends(nPrimaryKey); // 사용자에게 목록 출력
}

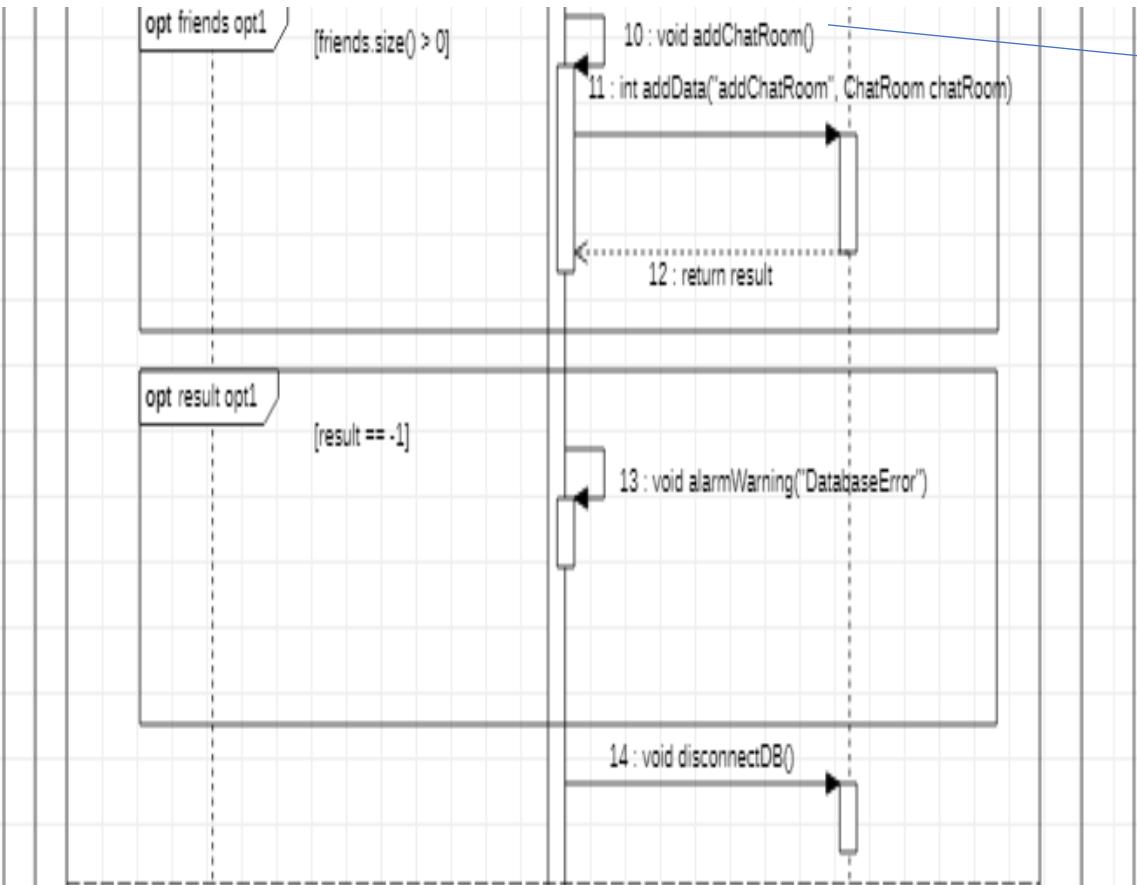
public void setKeyword(String strKeyword) {
    this.strKeyword = strKeyword;
}

public List<Friend> getUserFriends(int nPrimaryKey) {
    List<Friend> list = new ArrayList<Friend>();
    Friend friend = new Friend();
    list.add(friend); // DB에서 가져왔다고 가정
    // 사용자에게 친구목록을 보여준다.
    return list;
}
  
```

Search 클래스

DbManager 클래스

## 시퀀스 다이어그램



## 구현 코드

```

else if(nChatOption != 1) { //일반채팅
    showFriend(); //초대대상선택페런스
    if(friends.size() > 0) {
        addChatRoom();
        db.disconnectDB();
    }
} else{ //오픈채팅
    showOpenChatRooms();
    makeOpenChatRoom();
}

public void addChatRoom() {
    int result;
    result=db.addData("addChatRoom", chatroom);
    if(result == -1)
        alarmWarning("DatabaseError");
}

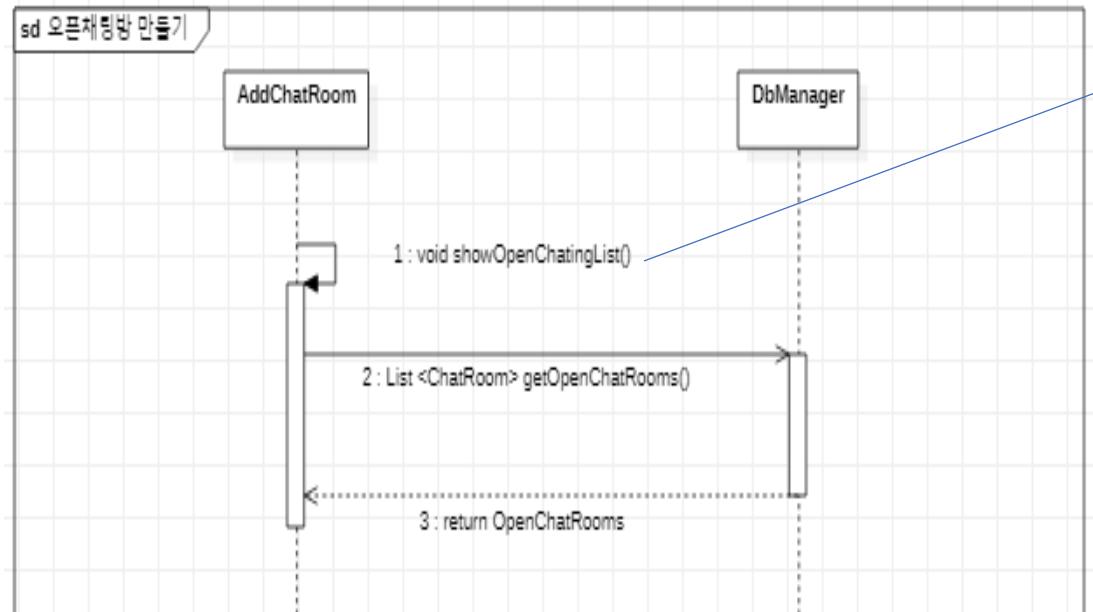
public int addData(String option, Object Data) {
    int n=0;
    //데이터삽입
    return n;
}
  
```

AddChatRoom의  
makeChatRoom()

addChatRoom() 내에서  
opt를 구현한 이유:  
액티비에이션바 표기 오류

DbManager 클래스

## 시퀀스 다이어그램



## 구현 코드

```
else{ //오픈채팅
    showOpenChatRooms();
    makeOpenChatRoom();
}

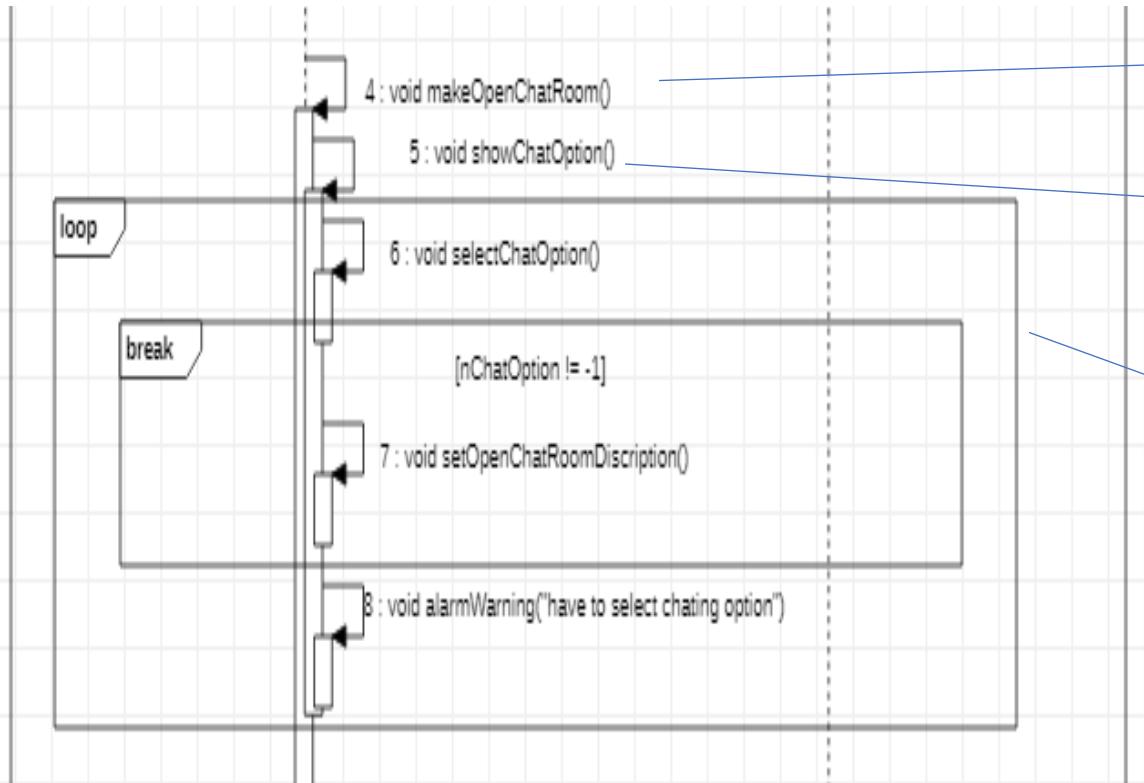
public void showOpenChatRooms() { //오픈채팅방 보여주기
    List<ChatRoom> openChatList = new ArrayList<ChatRoom>();
    openChatList = db.getOpenChatRooms();
}

public List<ChatRoom> getOpenChatRooms() { //오픈채팅방리스트 반환
    List<ChatRoom> list = new ArrayList<ChatRoom>();
    return list;
}
```

The implementation code corresponds to the sequence diagram. It shows the logic for handling open chat rooms. It includes an else block for open chat rooms, calling the `showOpenChatRooms()` and `makeOpenChatRoom()` methods. The `showOpenChatRooms()` method initializes a list and calls the `getOpenChatRooms()` method from the `DbManager` class. The `getOpenChatRooms()` method returns a list of chat rooms.

DbManager 클래스

## 시퀀스 다이어그램



## 구현 코드

```

else{//오픈채팅
    showOpenChatRooms();
    makeOpenChatRoom();
}

public void makeOpenChatRoom() {
    showOpenChatOption();
}

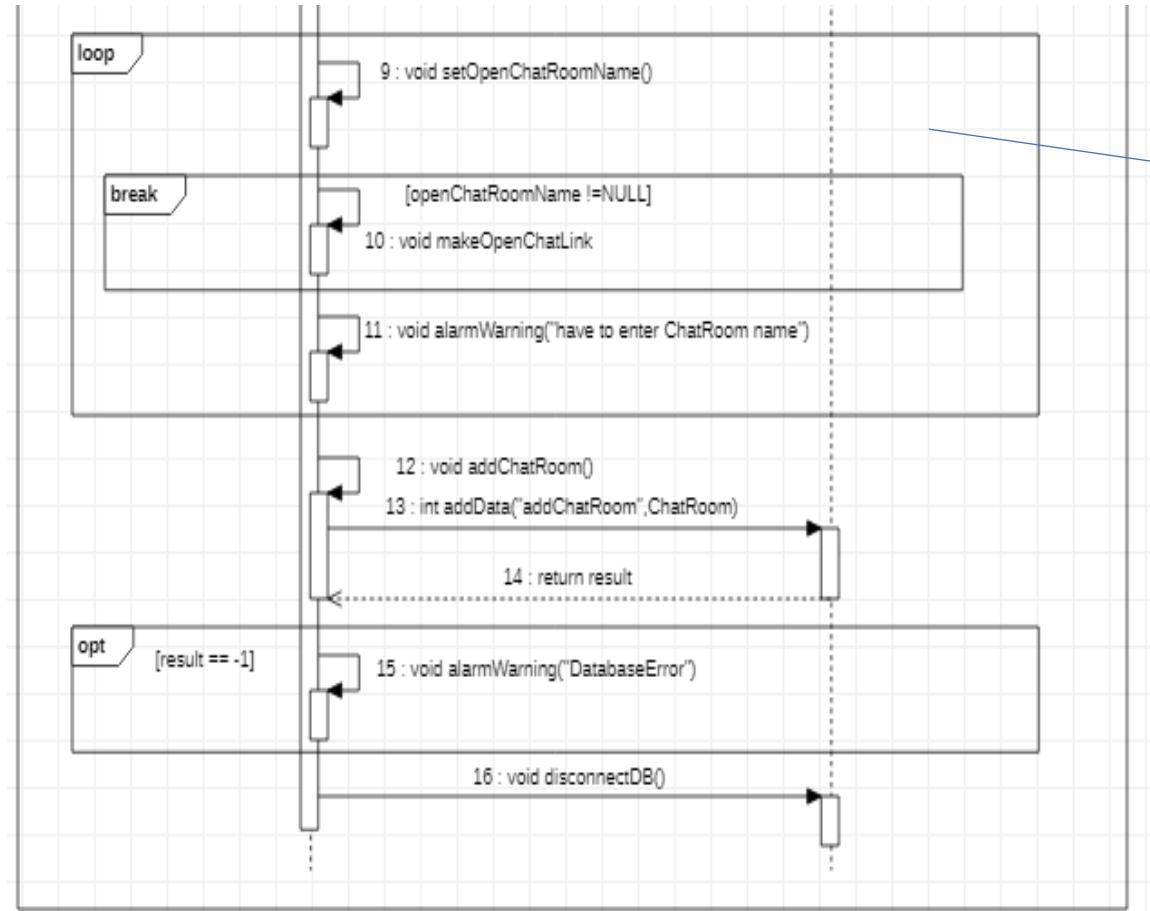
public void showOpenChatOption() {
    while(true) {
        selectChatOption();
        if(nChatOption!=-1) {
            setOpenChatDiscription();
            break;
        }
        alarmWarning("have to select chating option");
    }
}

public void setOpenChatDiscription() {//UI에 입력된 채팅방설명을 chatRoom에 저장
    String str= "채팅방설명";
    chatroom.chatRoomDiscription=str;
}

```

showOpenChatOption() 추가 이유 : 시퀀스의 흐름이 showChatOption()과 다르고 보여줄 옵션이 다르기 때문

## 시퀀스 다이어그램



## 구현 코드

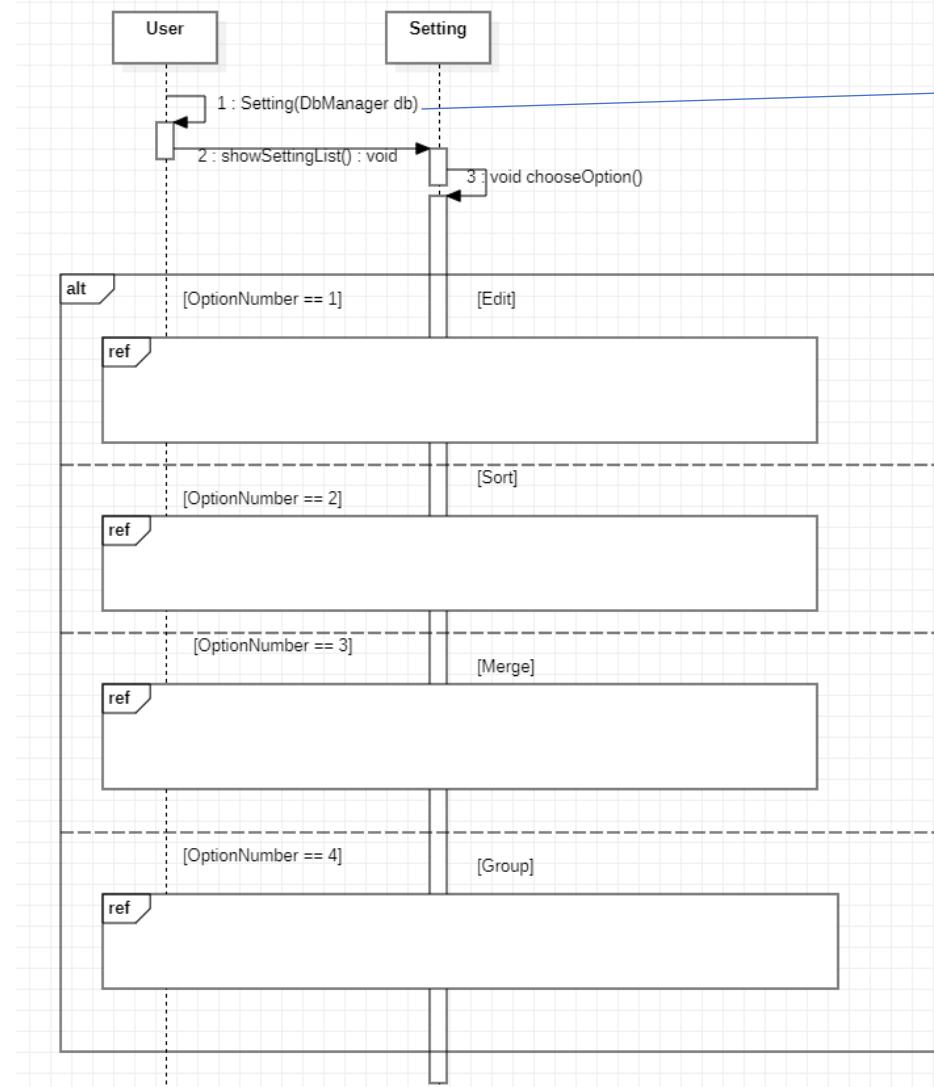
```

public void makeOpenChatRoom() {
    showOpenChatOption();
    while(true) {
        if(getOpenChatName() !=null) {
            chatroom.strChatRoomName=getOpenChatName();
            makeOpenChatLink();
            break;
        }
        alarmWarning("have to enter ChatRoom name");
    }
    addChatRoom();
    db.disconnectDB();
}

public String getOpenChatName() { //UI에 입력된 채팅방이름을 가져옴
    String str="채팅방이름";
    return str;
}

public void makeOpenChatLink() {
    String str="채팅방링크";
    chatroom.strOpenChatLink=str;
}
  
```

## 시퀀스 다이어그램



## 구현 코드

```

class User {
    void Setting(DbManager db) {
        System.out.println("Setting");
        Setting setting = new Setting(db);
        setting.showSettingList();
    }
}

public class ServerManagement2 {
    private static DbManager db;

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        User user = new User();
        user.Setting(db);
    }
}

public void showSettingList() {
    System.out.println("showSettingList");
    while(true) {
        System.out.println("1. Edit");
        System.out.println("2. Sort");
        System.out.println("3. Group");
        System.out.println("4. Merge");
        System.out.println("5. exit");
        chooseOption();
    }
}

public void chooseoption() {
    System.out.println("chooseOption");
    System.out.print(">>");
    int OptionNumber = sc.nextInt();

    if(OptionNumber == 1)
    {
        EditChatRoom(); //편집(1)
    }
    else if(OptionNumber == 2)
    {
        sortChatRoom(); //정렬(1)
    }
}

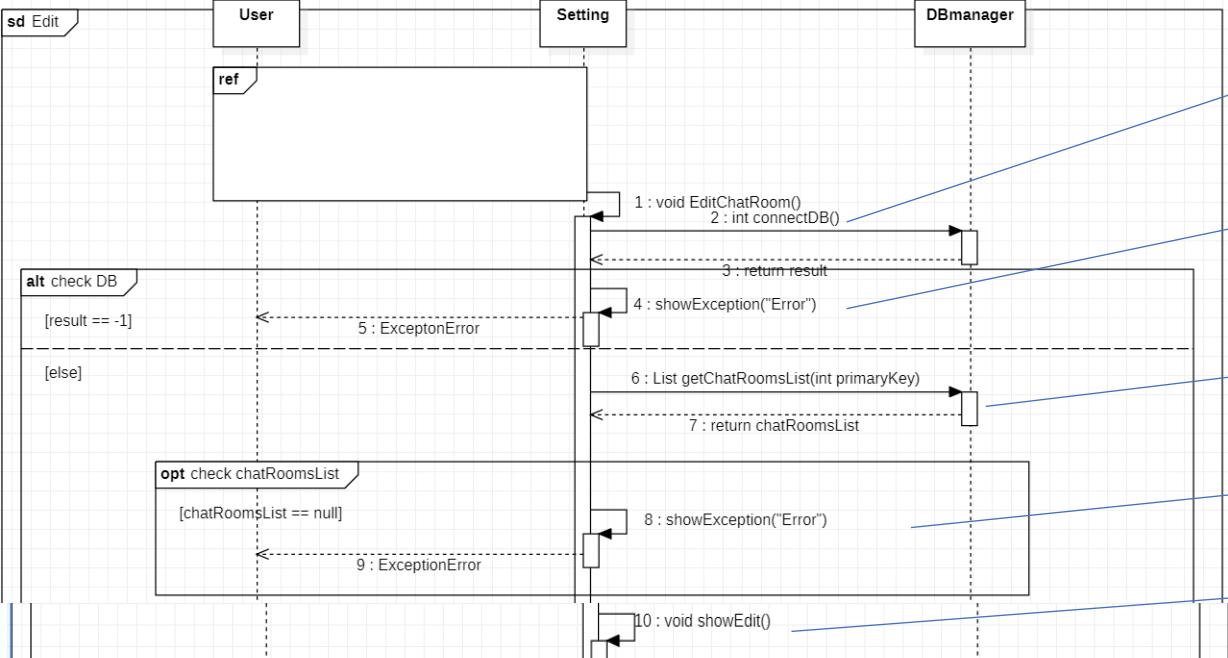
```

메인클래스에서 user.Setting(db) 실행

Setting 클래스

Setting 클래스

## 시퀀스 다이어그램

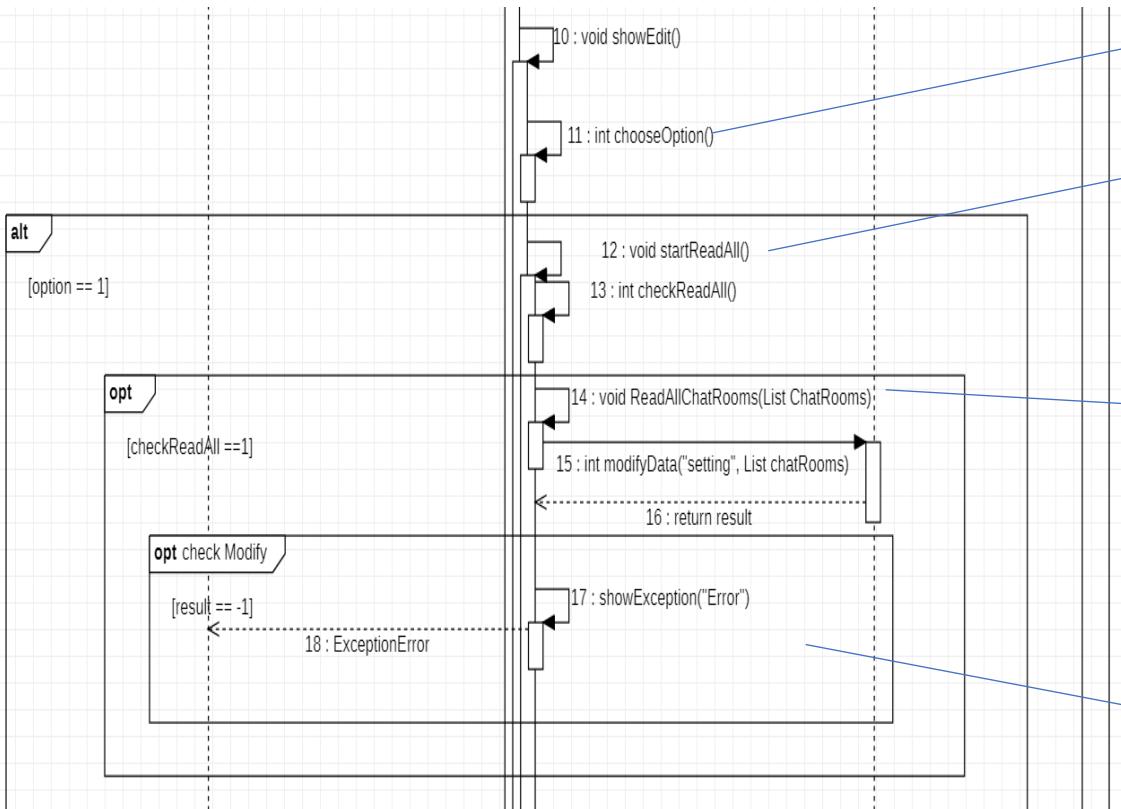


## 구현 코드

```

public void EditChatRoom() { //추가
    System.out.println("채팅방 편집");
    int result = db.connectDB(); //편집(2)(3) : 데이터베이스연결
    if(result == -1) //db접근에 이상이 있을경우
    {
        showException("Error"); //편집(4) : 데이터베이스 문제확인
        System.exit(0); //ExceptionError 편집(5)
    }
    else //db접근에 이상이 없을경우
    {
        List chatRoomsList = db.getChatRooms(nPrimaryKey); // 편집(6)(7)
        if(chatRoomsList == null) // null이면 에러
        {
            showException("Error"); //편집(8)
            System.exit(0); //ExceptionError 편집(9)
        }
        showEdit(); //편집(10)
    }
}
  
```

## 시퀀스 다이어그램



## 구현 코드

```

showEdit()

int option = chooseOption(); //편집 (11) : 편집기능 선택
if(option == 1) //모두읽음 기능
{
    startReadAll(); //편집 (12) : 모두 읽기 기능 시작
}

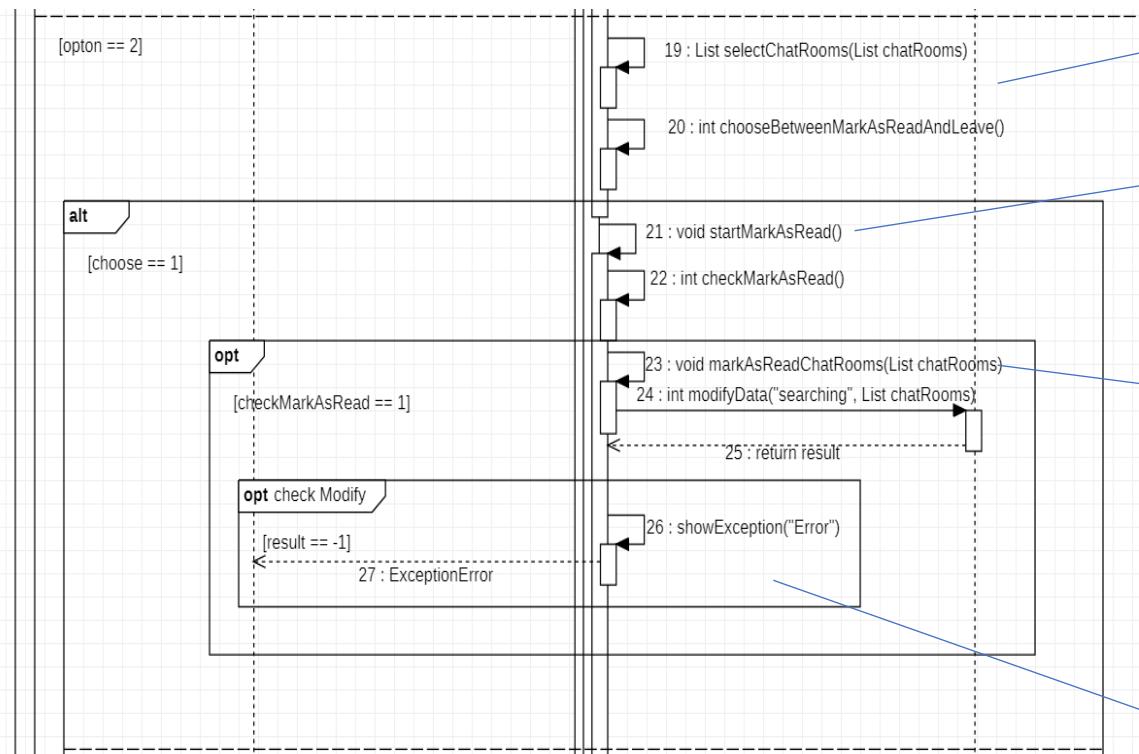
public void startReadAll() {
    int checkReadAll = checkReadAll(); //편집 (13) : 모두읽음을 동작할것인지 재확인
    if(checkReadAll == 1)
    {
        readAllChatRooms(ChatRooms); //편집 (14) : 모두읽기
    }
}

public void readAllChatRooms(List chatRooms) {
    int checkModify = db.modifyData("setting", chatRooms); // 편집(15)(16)
    if(checkModify == -1)
    {
        showException("Error"); //편집(17)
        System.exit(0); //ExceptionError 편집(18)
    }
}

```

The implementation code corresponds to the sequence diagram. It starts with `showEdit()`. It then checks if `option == 1` (편집기능 선택). If true, it calls `startReadAll()` (모두 읽기 기능 시작). The `startReadAll()` method calls `checkReadAll()` (모두읽음을 동작할것인지 재확인) and then `readAllChatRooms(ChatRooms)` (모두읽기). The `readAllChatRooms()` method calls `db.modifyData("setting", chatRooms)` (편집(15)(16)). If the result is -1, it calls `showException("Error")` (편집(17)) and exits the program (`System.exit(0)`) (ExceptionError 편집(18)).

## 시퀀스 다이어그램



## 구현 코드

```

showEdit()

else if(option == 2)
{
    selectChatRooms(ChatRooms); //편집(19)
    int choose = chooseBetweenMarkAsReadAndLeave(); //편집(20)
    if(choose == 1)
    {
        startMarkAsRead(); //편집(21)
    }

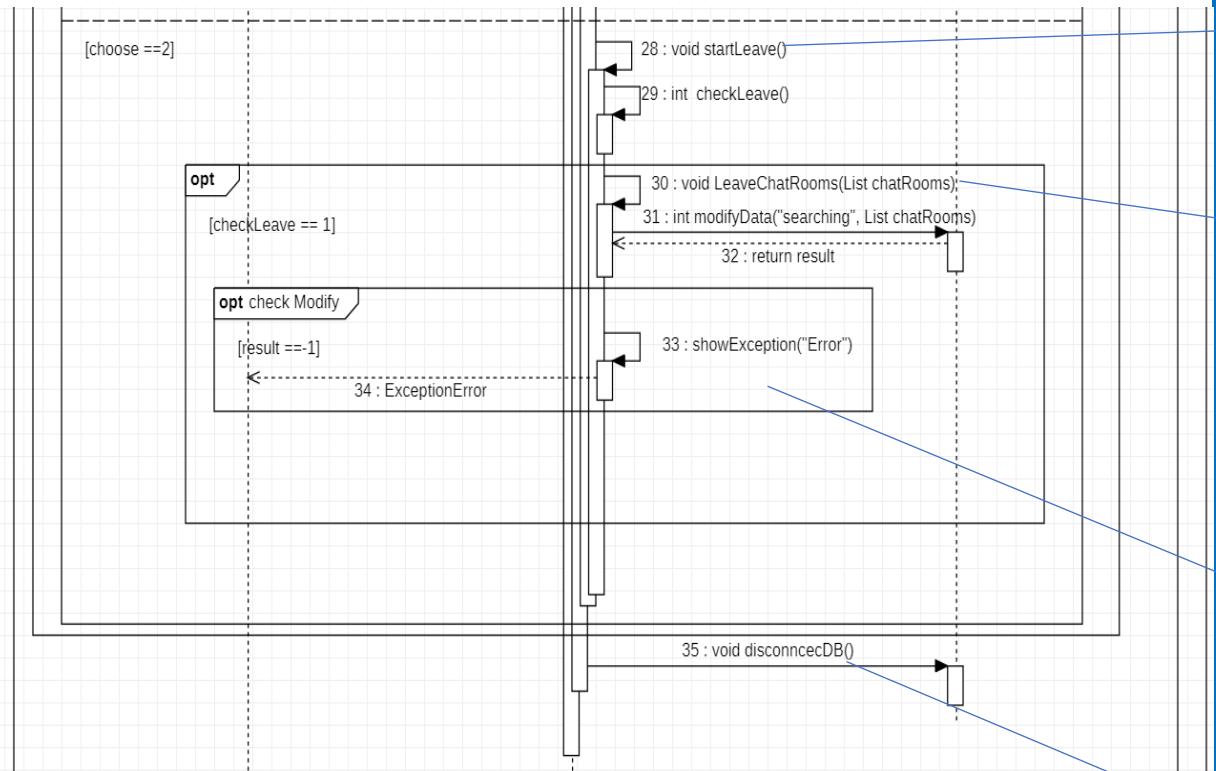
    public void startMarkAsRead() {
        int checkMarkAsRead = checkMarkAsRead(); //편집(22)
        if(checkMarkAsRead == 1)
        {
            markAsReadChatRooms(ChatRooms); //편집(23)
        }
    }

    public void markAsReadChatRooms(List chatRooms) {
        int checkModify = db.modifyData("searching", chatRooms); //편집(24)(25)
        if(checkModify == -1)
        {
            showException("Error");//편집(26)
            System.exit(0); //ExceptionError 편집(27)
        }
    }
}

```

The implementation code corresponds to the sequence diagram. It begins with `showEdit` and handles the selection of option 2. It calls `selectChatRooms` and `chooseBetweenMarkAsRead`. If the choice is 1, it executes `startMarkAsRead`. This method calls `checkMarkAsRead` and then `markAsReadChatRooms`. Finally, it calls `modifyData` with the parameter "searching". If `checkModify` is -1, it shows an error message and exits the application. The code uses red boxes to highlight specific parts of the implementation that correspond to the numbered steps in the sequence diagram.

## 시퀀스 다이어그램



## 구현 코드

```

showEdit()

else if(choose == 2)
{
    startLeave(); //편집(28)
}

public void startLeave() {
    int checkLeave = checkLeave(); //편집(29)
    if(checkLeave == 1)
    {
        LeaveChatRooms(ChatRooms); //편집(30)
    }
}

public int checkLeave() {
    int n=0;
    return n;
}

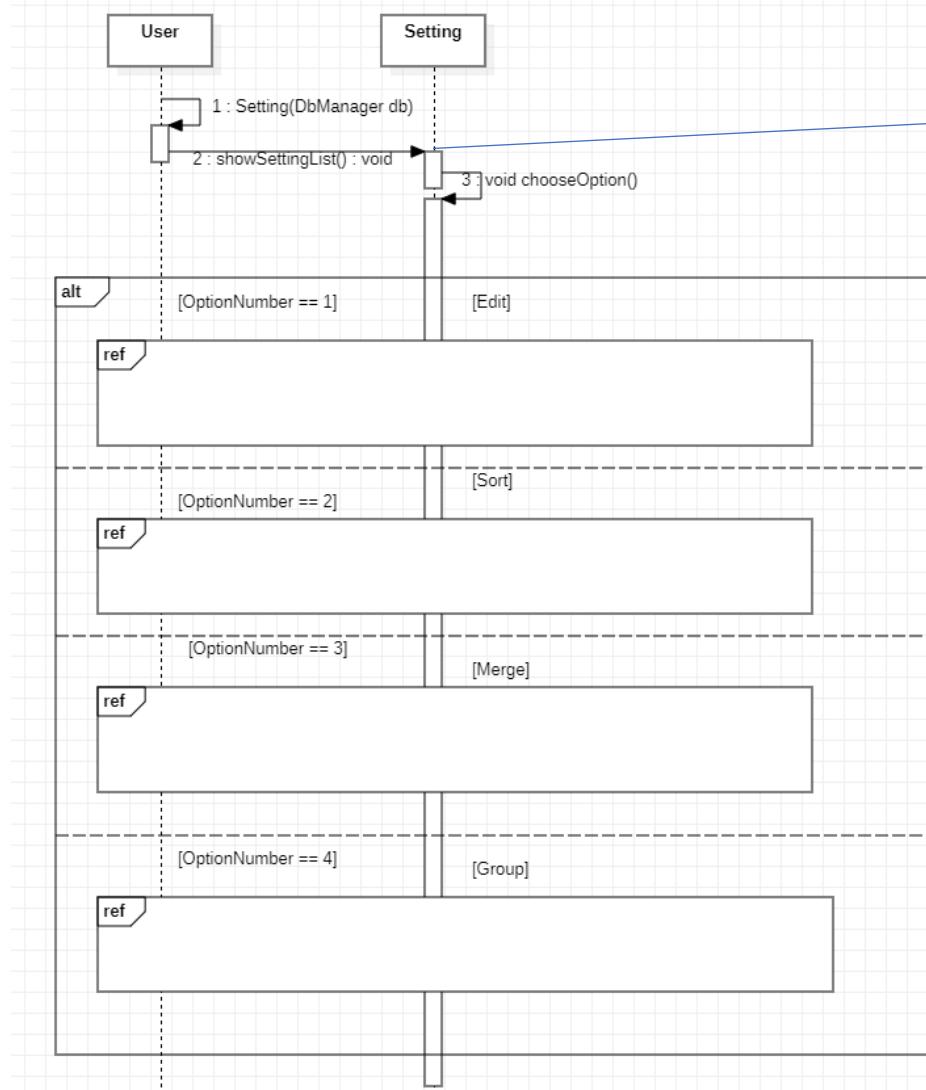
public void LeaveChatRooms(List chatRooms) {
    int checkModify = db.modifyData("searching", chatRooms); //편집(31)(32)
    if(checkModify == -1)
    {
        showException("Error"); //편집(33)
        System.exit(0); //ExceptionError 편집(34)
    }
}

showEdit()

db.disconnectDB(); //편집(35)

```

## 시퀀스 다이어그램



## 구현 코드

```

class User{
    void Setting(DbManager db) {
        System.out.println("Setting");
        Setting setting = new Setting(db);
        setting.showSettingList();
    }
}

public class ServerManagement2 {
    private static DbManager db;

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        User user = new User();
        user.Setting(db);
    }
}

Setting 클래스
public void showSettingList() {
    System.out.println("showSettingList");
    while(true) {
        System.out.println("1. Edit");
        System.out.println("2. Sort");
        System.out.println("3. Group");
        System.out.println("4. Merge");
        System.out.println("5. exit");
        chooseoption();
    }
}

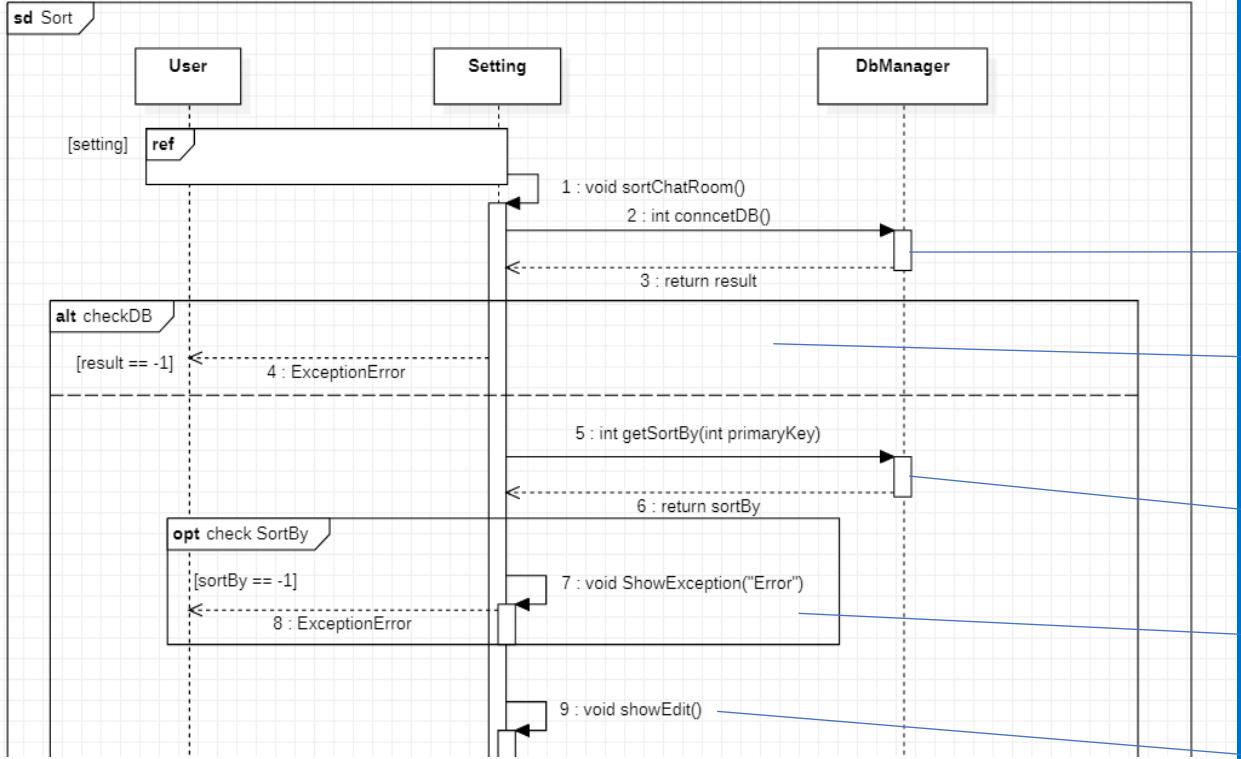
Setting 클래스
public void chooseoption() {
    System.out.println("chooseOption");
    System.out.print(">>");
    int OptionNumber = sc.nextInt();

    if(OptionNumber == 1)
    {
        EditChatRoom(); //편집(1)
    }
    else if(OptionNumber == 2)
    {
        sortChatRoom(); //정렬(1)
    }
}

```

The implementation code shows the User and Setting classes. The User class has a constructor that prints "Setting", creates a new Setting object, and calls its showSettingList() method. The main method of the ServerManagement2 class creates a User object and calls its Setting(db) method. The Setting class contains a showSettingList() method that prints various options and enters a loop. The chooseoption() method prints "chooseOption", reads the user's input, and performs actions based on the input: if OptionNumber is 1, it calls EditChatRoom(); if it is 2, it calls sortChatRoom(). The code uses System.out.println() for output and Scanner.nextInt() for input.

## 시퀀스 다이어그램



## 구현 코드

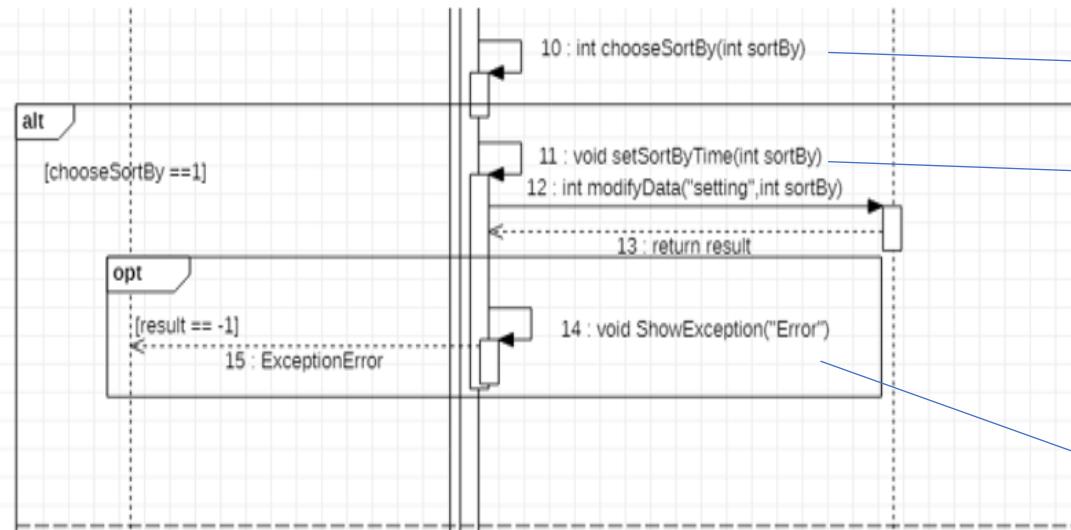
```

public void sortChatRoom() {
    int result = db.connectDB(); //정렬(2)(3)
    if(result == -1)
    {
        System.exit(0); //ExceptionError 정렬(4)
    }
    else
    {
        int checkSortBy = db.getSortBy(nPrimaryKey); //정렬(5)(6)
        if(checkSortBy == -1)
        {
            showException("Error"); //정렬(7)
            System.exit(0); //ExceptionError 정렬(8)
        }
        showSort(); //정렬(9)
        db.disconnectDB(); //정렬(26)
    }
}

```

showSort()를 추가한 이유 : 편집과 정렬은 다른 메소드가 필요하기 때문

## 시퀀스 다이어그램



## 구현 코드

```

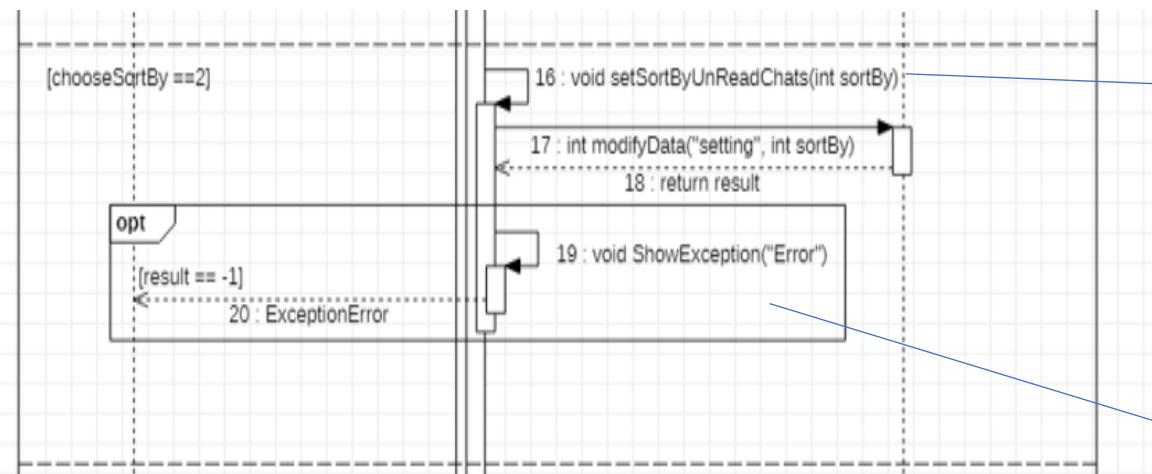
public void showSort() {
    System.out.println("정렬기능을 보여준다");
    int chooseSortBy = chooseSortBy(nSortBy); //정렬(10)
    if(chooseSortBy == 1) //최근메시지 순
    {
        setSortByTime(nSortBy); //정렬(11)
    }
}

public void setSortByTime(int sortBy)
{
    int result = db.modifyData("setting", sortBy); //정렬(12)(13)
    if(result == -1)
    {
        showException("Error"); //정렬(14)
        System.exit(0); //ExceptionError 정렬(15)
    }
}

```

The implementation code shows the mapping of the sequence steps to Java code. The `showSort()` method prints a message and calls `chooseSortBy(nSortBy)` (line 10). It then checks if `chooseSortBy == 1` and calls `setSortByTime(nSortBy)` (line 11). The `setSortByTime(int sortBy)` method calls `db.modifyData("setting", sortBy)` (line 13). If the result is -1, it calls `showException("Error")` (line 14) and exits the program (line 15).

## 시퀀스 다이어그램



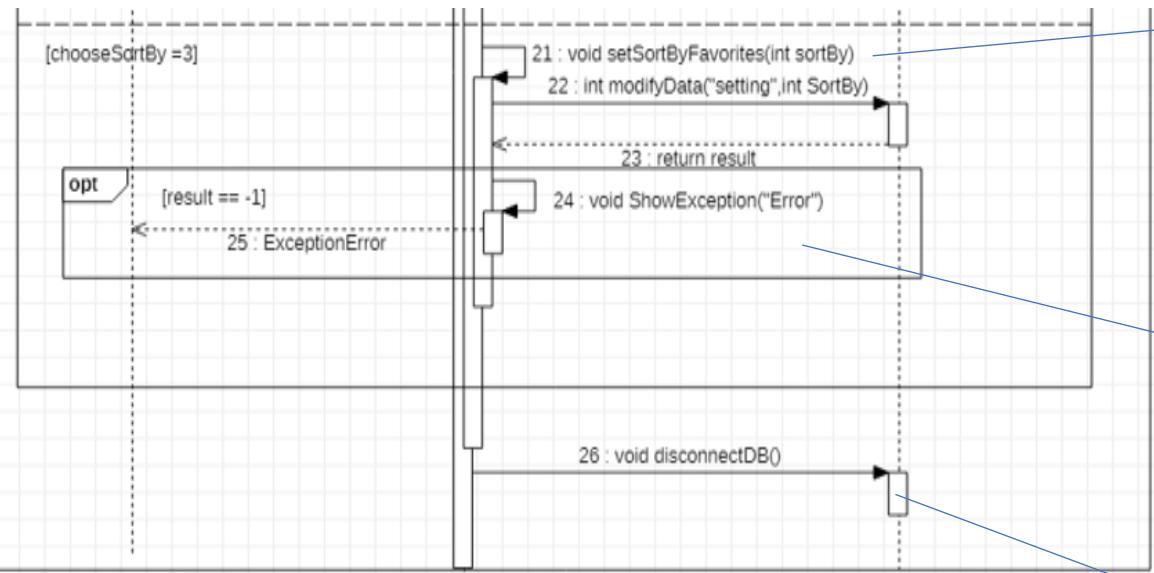
## 구현 코드

```
showSort()

else if(chooseSortBy == 2) //안읽은 메시지 순
{
    → setSortByUnReadChats(nSortBy); //정렬(16)
}

public void setSortByUnReadChats(int sortBy)
{
    int result = db.modifyData("setting", sortBy); //정렬(17)(18)
    if(result == -1)
    {
        → showException("Error"); //정렬(19)
        System.exit(0); //ExceptionError 정렬(20)
    }
}
```

## 시퀀스 다이어그램



## 구현 코드

```

showSort()

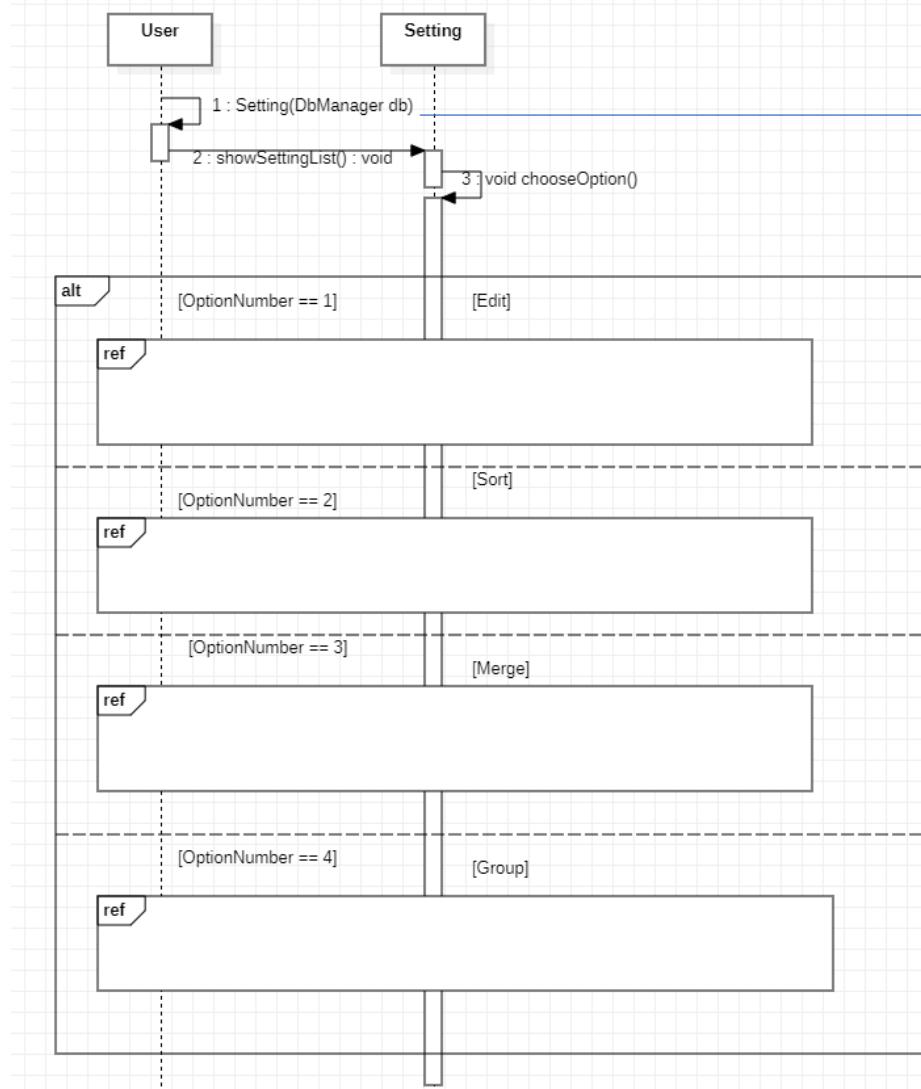
else if(chooseSortBy == 3)// 즐겨찾기순
{
    setSortByFavorites(nSortBy); //정렬(21)
}

public void setSortByFavorites(int sortBy)
{
    int result = db.modifyData("setting", sortBy); //정렬(22)(23)
    if(result == -1)
    {
        showException("Error"); //정렬(24)
        System.exit(0); //ExceptionError 정렬(25)
    }
}

showSort()

db.disconnectDB(); //정렬(26)
  
```

## 시퀀스 다이어그램



## 구현 코드

```

class User{
    void Setting(DbManager db) {
        System.out.println("Setting");
        Setting setting = new Setting(db);
        setting.showSettingList();
    }
}

public class ServerManagement2 {
    private static DbManager db;

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        User user = new User();
        user.Setting(db);
    }
}

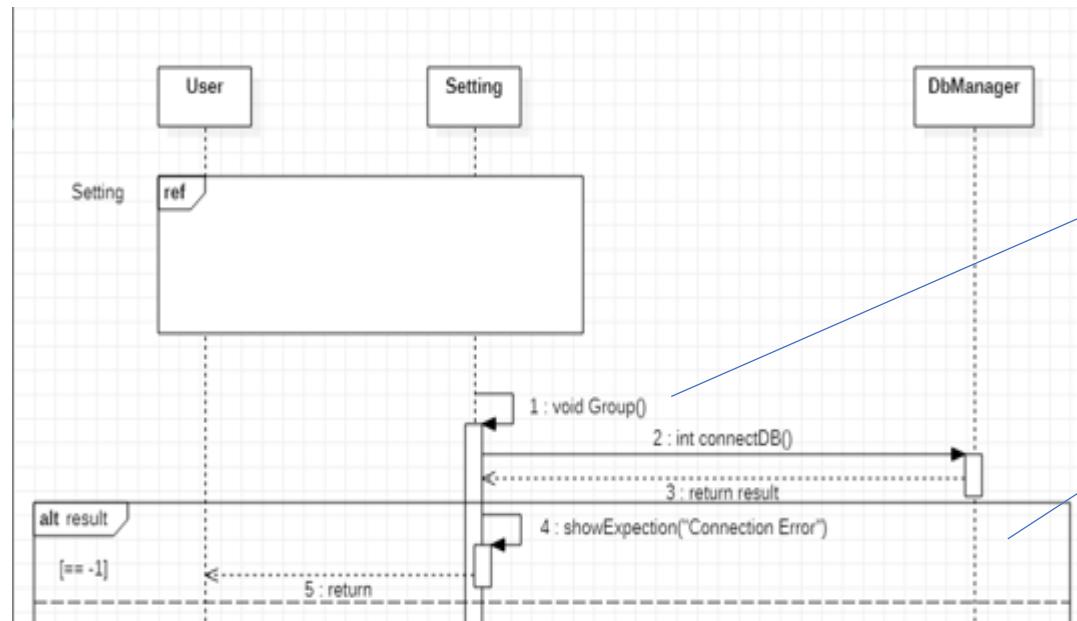
Setting 클래스
public void showSettingList() {
    System.out.println("showSettingList");
    while(true) {
        System.out.println("1. Edit");
        System.out.println("2. Sort");
        System.out.println("3. Group");
        System.out.println("4. Merge");
        System.out.println("5. exit");
        chooseOption();
    }
}

Setting 클래스
public void chooseOption() {
    System.out.println("chooseOption");
    System.out.print(">>");
    int OptionNumber = sc.nextInt();
    if(OptionNumber == 3) {
        Group();
    } else if(OptionNumber == 4) {
        mergeChatRoom();
    } else if(OptionNumber == 5) {
        System.exit(0);
    }
}

```

The implementation code shows the User class containing a Setting method that prints "Setting" and creates a Setting object with the provided db manager. The main method creates a User object and calls its Setting method. The Setting class has a showSettingList() method that prints "showSettingList" and enters a loop. Inside the loop, it prints five options (1. Edit, 2. Sort, 3. Group, 4. Merge, 5. exit) and calls the chooseOption() method. The chooseOption() method prints "chooseOption", takes input from the user, and based on the input (1, 2, or 5), it calls either the Group(), mergeChatRoom(), or System.exit(0) methods respectively.

## 시퀀스 다이어그램



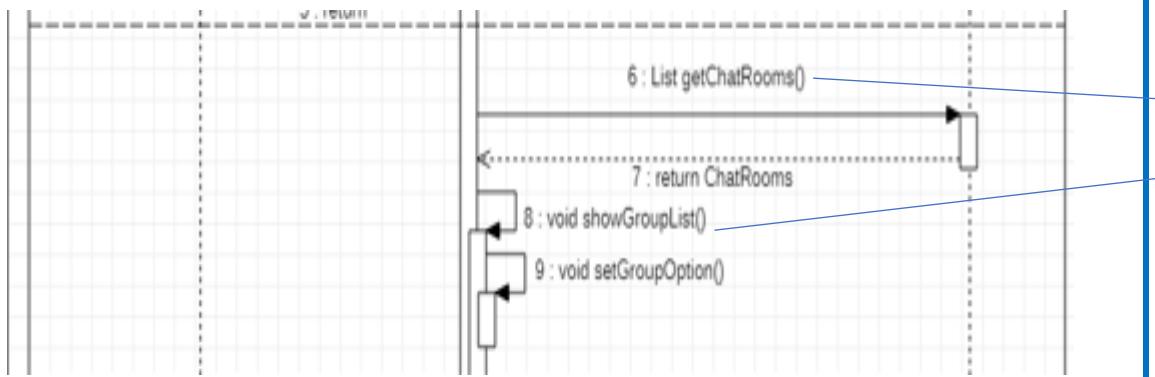
## 구현 코드

```

public void Group() {
    System.out.println("Group");
    db = new DbManager();
    ChatRooms = new ArrayList<ChatRoom>();
    int result = db.connectDB();
    if(result == -1) {
        showException("Connection Error");
    } else {
        ChatRooms = db.getChatRooms(nPrimaryKey); //여러 개의 채팅방을 들고와야 할
        showGroupList();
    }
}

public void showException(String str) {
    System.out.println("showException");
    System.out.println(str);
    System.exit(0);
}
  
```

## 시퀀스 다이어그램



## 구현 코드

```

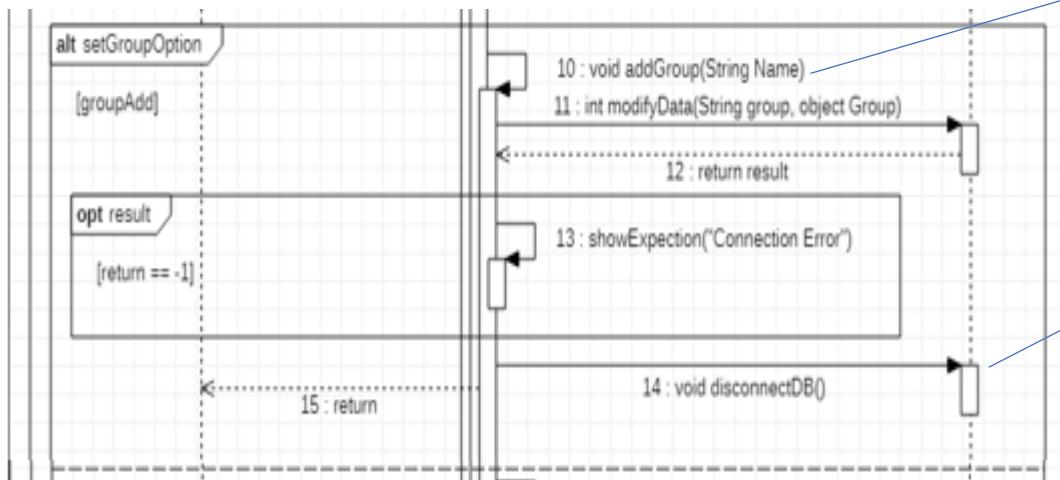
public void Group() {
    System.out.println("Group");
    db = new DbManager();
    ChatRooms = new ArrayList<ChatRoom>();
    int result = db.connectDB();
    if(result == -1) {
        showException("Connection Error");
    } else {
        ChatRooms = db.getChatRooms(nPrimaryKey); //여러 개의 채팅방을 들고와야 함
        showGroupList();
    }
}

List getChatRooms(int nPrimaryKey) {
    System.out.println("getChatRooms");
    // nPrimaryKey에 해당하는 List 반환
    return null;
}

public void showGroupList() {
    System.out.println("showGroupList");
    Group group = new Group();
    System.out.println("group num : "+group.nNum);
    System.out.println("group name : "+group.strName);
    setGroupOption();
    if(nooption == 1) {
        addGroup(group.strName);
    } else if(nooption == 2) {
        delGroup(group);
    }
}
  
```

DbManager 클래스

## 시퀀스 다이어그램



## 구현 코드

```

public void addGroup(String Name) {
    System.out.println("addGroup");
    Group group = new Group();
    int result;
    result = db.modifyData(group.strName, group);
    if(result == -1) {
        showException("Connection Error");
    }
    db.disconnectDB();
    System.exit(0);
}
  
```

DbManager 클래스

```

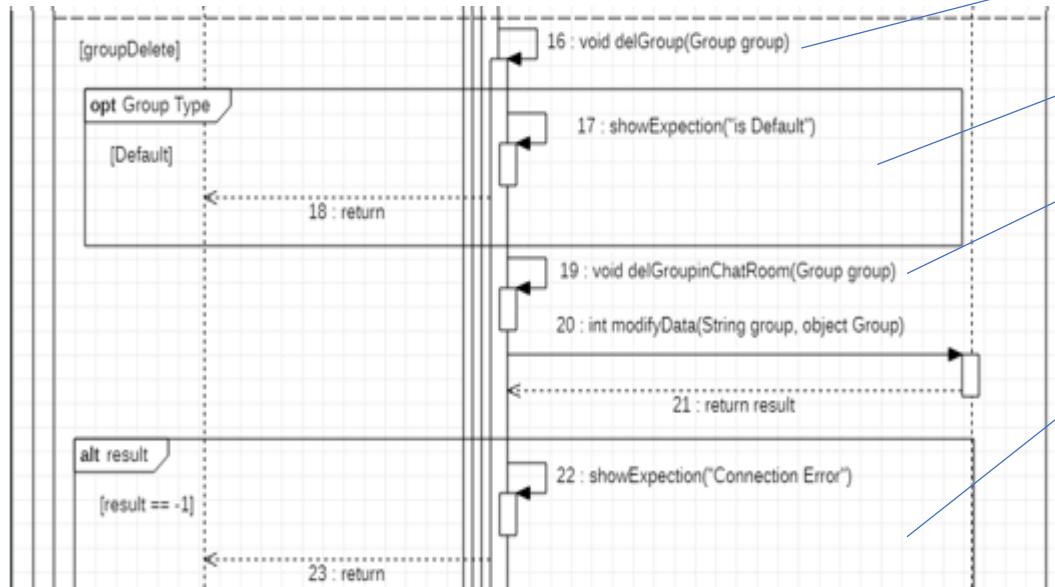
public int modifyData(String strName, Group group) {
    System.out.println("modifyData");
    //if(modifyData complete)
    //    return 0;
    //else
    //    return -1;
}
  
```

```

public void disconnectDB() {
    System.out.println("disconnectDB");
    // DB와 연결 해제
}
  
```

DbManager 클래스

## 시퀀스 다이어그램



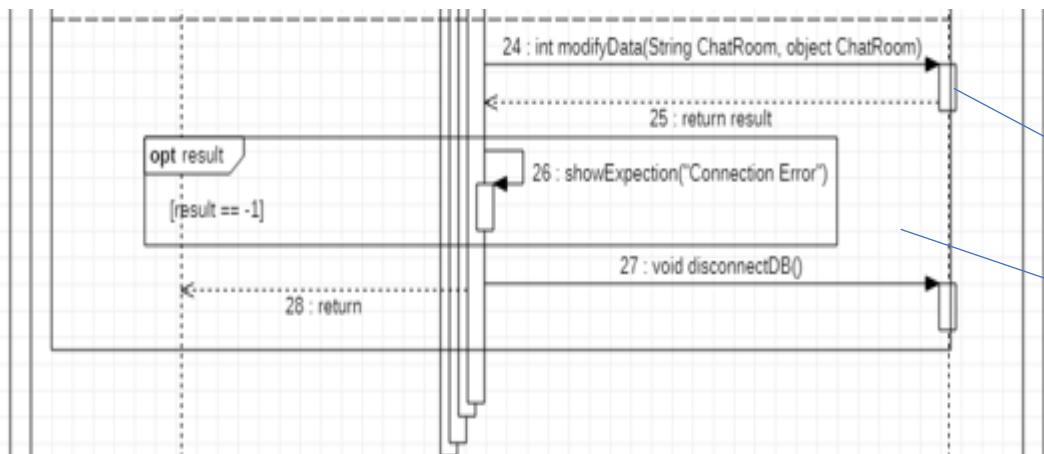
## 구현 코드

```

public void delGroup(Group group) {
    //if(group의 개수가 1개라면){
    //showException("is Default");
    //System.exit(0);
    //}
    delGroupinChatRoom(group);
    int result;
    result = db.modifyData(group.strName, group);
    if(result == -1) {
        showException("Connection Error");
        System.exit(0);
    }
    else {
        //result = db.modifyData(ChatRooms.get(0).strChatRoomName, group);
        if(result == -1) {
            showException("Connection Error");
        }
        db.disconnectDB();
        System.exit(0);
    }
}

public void delGroupinChatRoom(Group group) {
    //모든 채팅방의 그룹을 확인하여 파라미터로 받은 group을 삭제
}
  
```

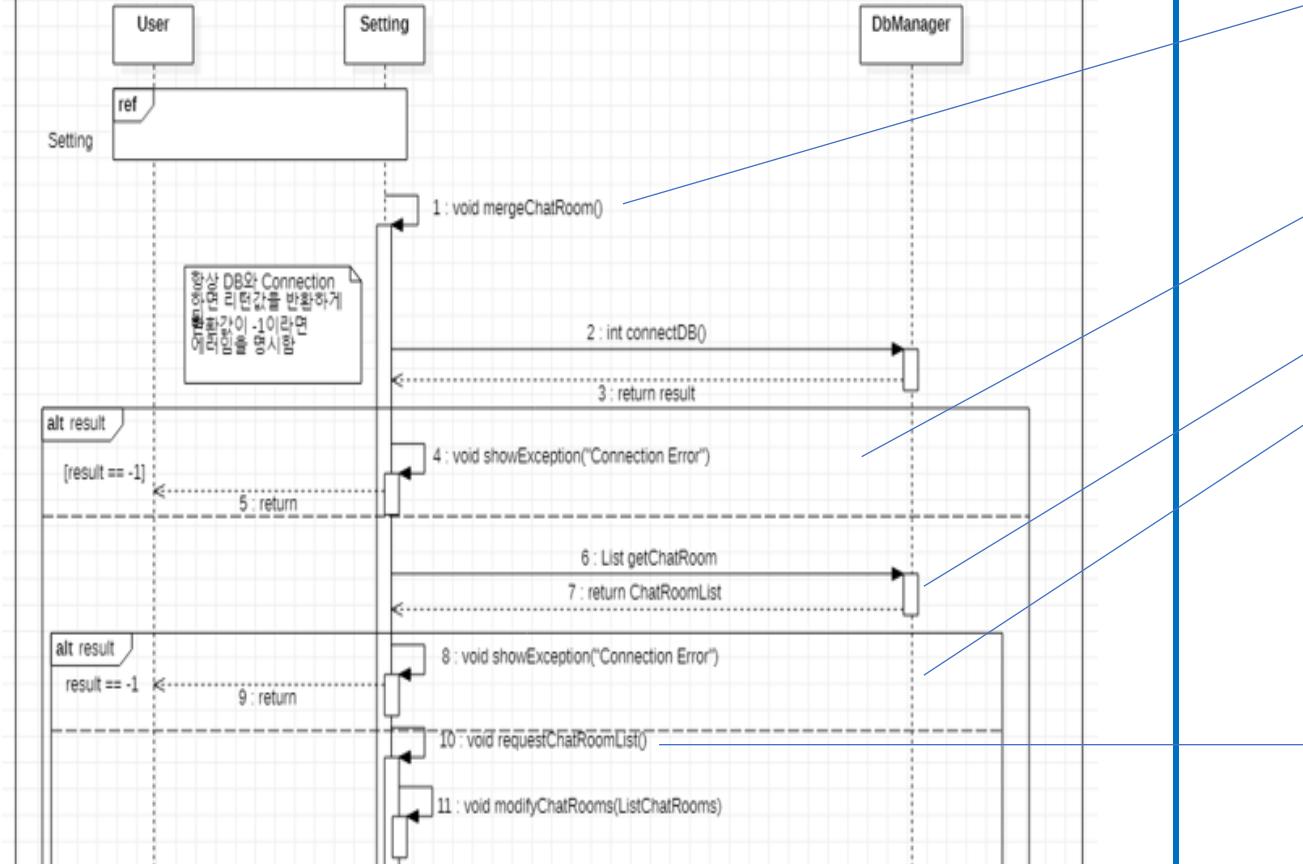
## 시퀀스 다이어그램



## 구현 코드

```
public void delGroup(Group group) {  
    //if(group의 개수가 1개라면){  
    //showException("is Default");  
    //System.exit(0);  
    //}  
    delGroupInChatRoom(group);  
    int result;  
    result = db.modifyData(group.strName, group);  
    if(result == -1) {  
        showException("Connection Error");  
        System.exit(0);  
    }  
    else {  
        //result = db.modifyData(ChatRooms.get(0).strChatRoomName, group);  
        if(result == -1) {  
            showException("Connection Error");  
        }  
        db.disconnectDB();  
        System.exit(0);  
    }  
}
```

## 시퀀스 다이어그램



## 구현 코드

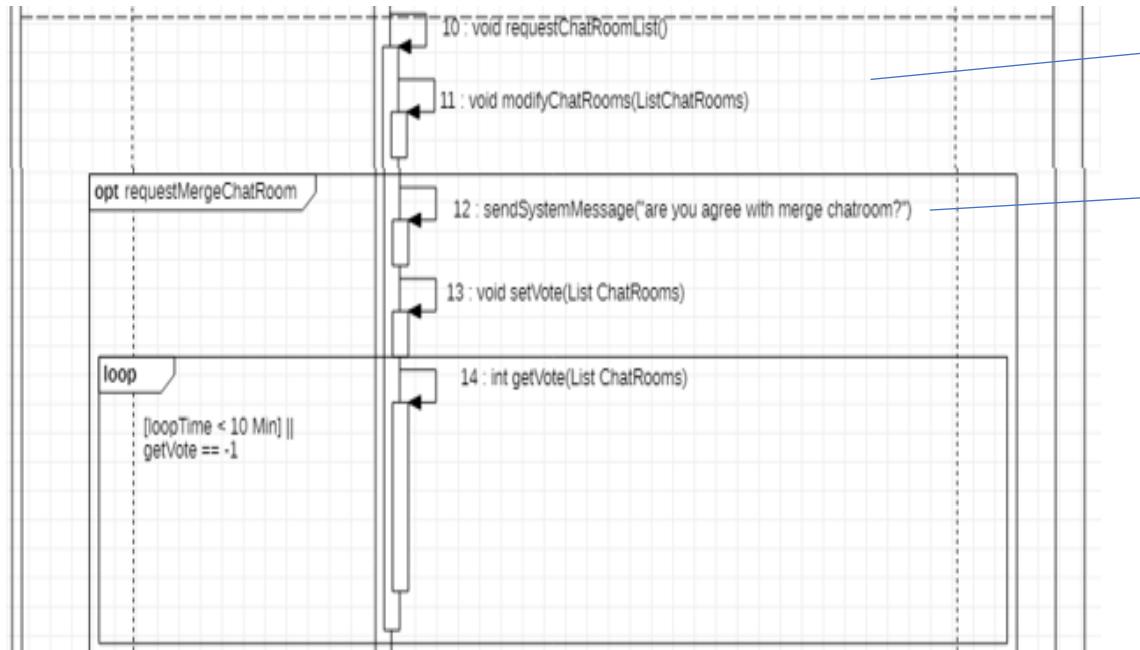
```

public void mergeChatRoom() {
    System.out.println("mergeChatRoom");
    db = new DbManager();
    ChatRooms = new ArrayList<ChatRoom>();
    int result = db.connectDB();
    if(result == -1) {
        showException("Connection Error");
    } else {
        ChatRooms = db.getChatRooms(nPrimaryKey);
        if(ChatRooms == null) {
            showException("Connection Error");
        } else {
            requestChatRoomList();
        }
        if(getVote(ChatRooms) != allVoteCounts) {
            db.disconnectDB();
            System.exit(0);
        }
        setMergeChat();
        result = 0;//createChatRoom(); deleteChatRoom(); modifydata();
        if(result == -1) {
            showException("Connection Error");
        } else {
            sendSystemMessage("Room Merged!");
            db.disconnectDB();
            System.exit(0);
        }
    }
}

```

The provided Java code implements the `mergeChatRoom()` method. It first prints "mergeChatRoom" and initializes `db`, `ChatRooms`, and `result`. It then calls `connectDB()` on `db`. If `result` is -1, it shows a connection error. Otherwise, it gets the ChatRooms from `db`. If the ChatRooms are null, it shows a connection error. Otherwise, it calls `requestChatRoomList()`. It then checks if the total vote count matches `allVoteCounts`. If not, it disconnects from the database and exits. It then sets the merge chat status, initializes `result` to 0 (representing no operation), and performs other database operations like creation, deletion, and modification. If any of these operations fail (`result == -1`), it shows a connection error. Otherwise, it sends a system message indicating the room was merged, disconnects from the database, and exits.

## 시퀀스 다이어그램



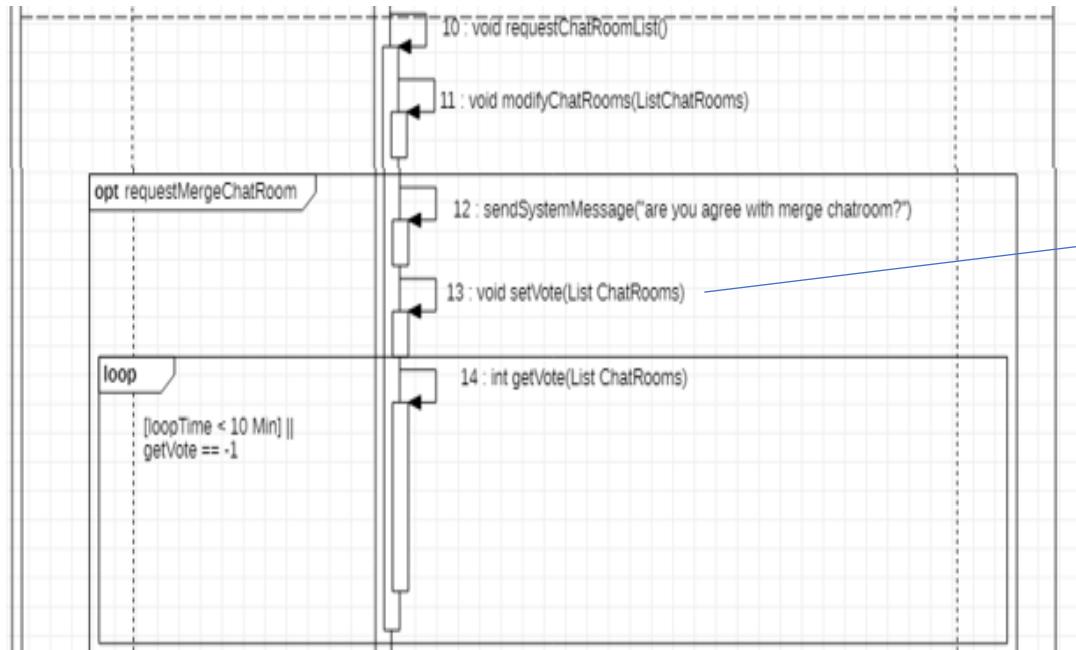
## 구현 코드

```
public void requestChatRoomList() {
    int loopTime = 0;
    System.out.println("requestChatRoomList");
    modifyChatRooms(ChatRooms);
    if(loopTime < 600 || getVote(ChatRooms) == -1 || getVote(ChatRooms) == 0) {
        sendSystemMessage("are you agree with merge chatroom?");
        setVote(ChatRooms);
        while(true) {
            if(loopTime < 600 || getVote(ChatRooms) == -1) {
                getVote(ChatRooms);
                break;
            }
        }
    }
}

public void modifyChatRooms(List<ChatRoom> list) {
    System.out.println("modifyChatRooms");
}

public void sendSystemMessage(String str) {
    System.out.println("sendSystemMessage");
    System.out.println(str);
}
```

## 시퀀스 다이어그램



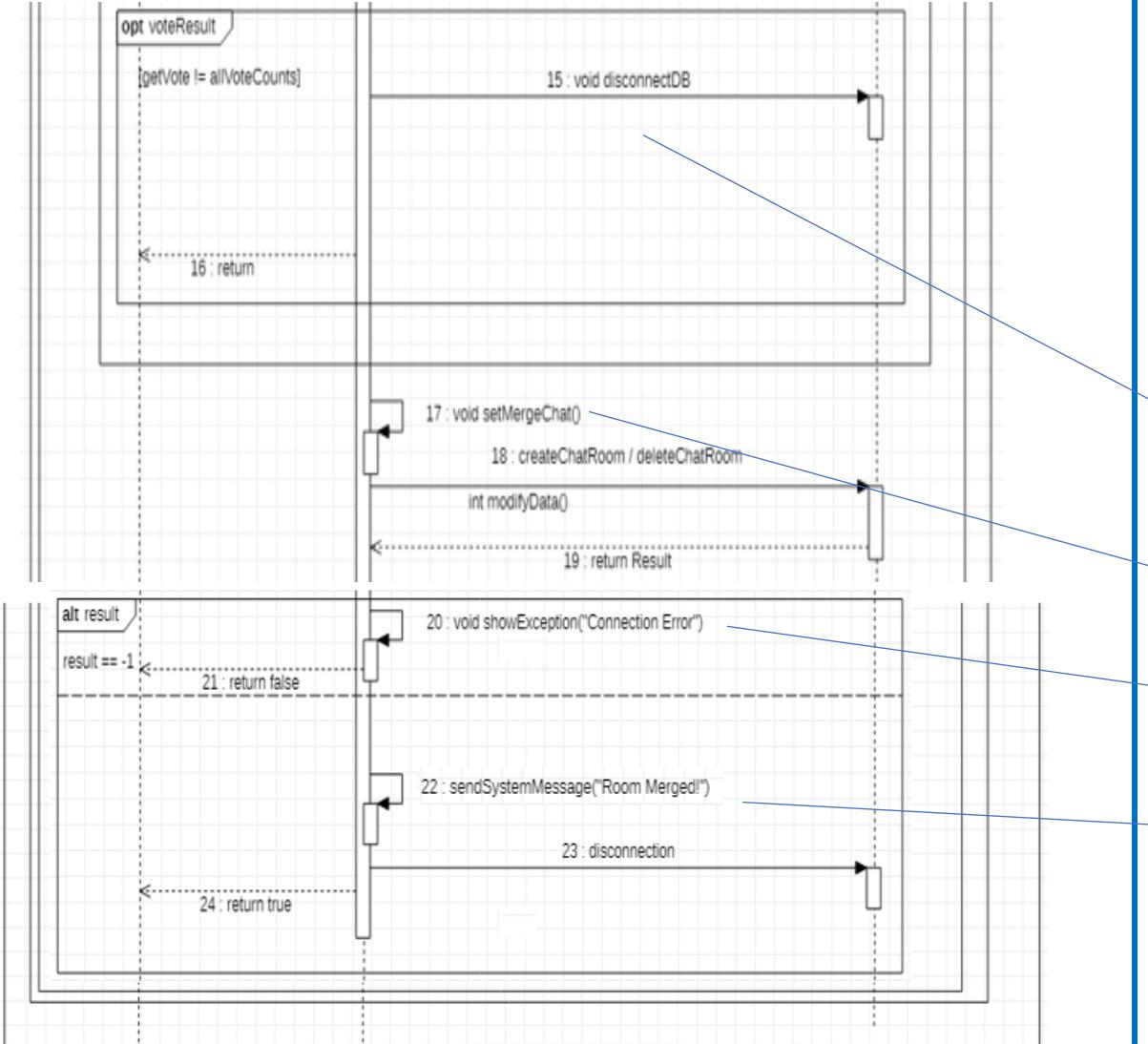
## 구현 코드

```
public void requestChatRoomList() {
    int loopTime = 0;
    System.out.println("requestChatRoomList");
    modifyChatRooms(ChatRooms);
    if(loopTime < 600 || getVote(ChatRooms) == -1 || getVote(ChatRooms) == 0) {
        sendSystemMessage("are you agree with merge chatroom?");
        setVote(ChatRooms);
        while(true) {
            if(loopTime < 600 || getVote(ChatRooms) == -1) {
                getVote(ChatRooms);
                break;
            }
        }
    }
}

public void setVote(List<ChatRoom> list) {
    System.out.println("setVote");
}

public int getVote(List<ChatRoom> list) {
    System.out.println("getVote");
    int n = 5; // n = 투표자 수를 나타내는 변수
    //if(투표 거절)
    //return -1;
    //else if (만장일치)
    //return 0;
    //else (투표)
    return n;
}
```

## 시퀀스 다이어그램



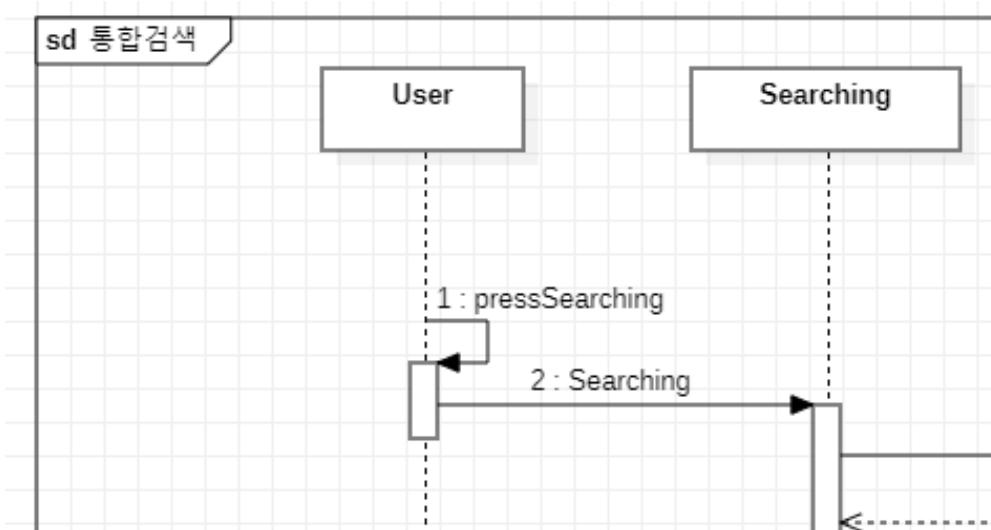
## 구현 코드

```

public void mergeChatRoom() {
    System.out.println("mergeChatRoom");
    db = new DbManager();
    ChatRooms = new ArrayList<ChatRoom>();
    int result = db.connectDB();
    if(result == -1) {
        showException("Connection Error");
    } else {
        ChatRooms = db.getChatRooms(nPrimaryKey);
        if(ChatRooms == null) {
            showException("Connection Error");
        } else {
            requestChatRoomList();
        }
        if(getVote(ChatRooms) != allVoteCounts) {
            db.disconnectDB();
            System.exit(0);
        }
        setMergeChat();
        result = 0;//createChatRoom(); deleteChatRoom(); modifydata();
        if(result == -1) {
            showException("Connection Error");
        } else {
            sendSystemMessage("Room Merged!");
            db.disconnectDB();
            System.exit(0);
        }
    }
}

```

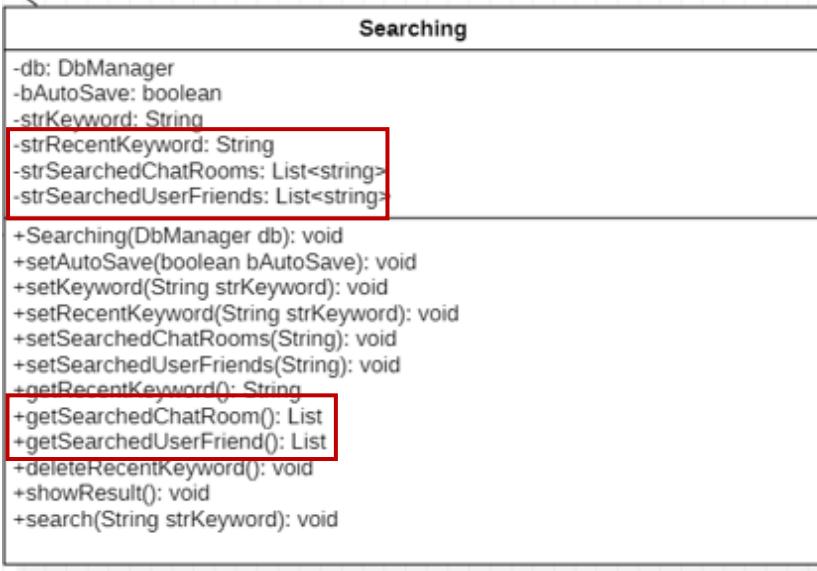
## ● 구현에 있어 어려웠던 점 : 실행 방식 Issue



User 클래스에서  
다른 클래스를 호출하여  
실행하게 되어있는 이슈

클래스들을 main에서  
호출하여 해결

## ● 구현에 있어 어려웠던 점 : 자료형 Issue

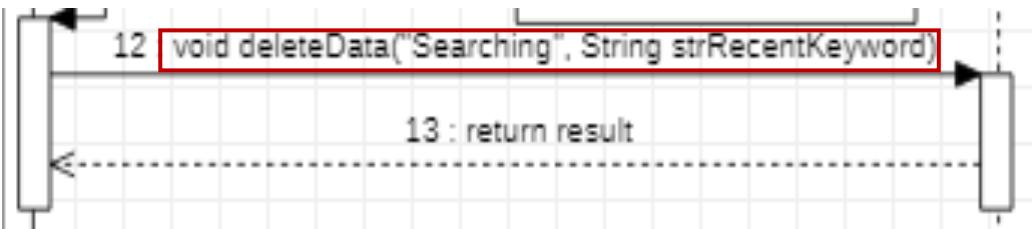


```
private ArrayList<String> strRecentKeyword = new ArrayList<String>();
private ArrayList<ChatRoom> SearchedChatRooms = new ArrayList<ChatRoom>();
private ArrayList<User> SearchedUserFriends = new ArrayList<User>();
```

클래스의 자료형이나  
반환형에 대한 이슈

각각의 필요에 따라 자료형과  
반환형을 바꿔주어 해결

## ● 구현에 있어 어려웠던 점 : 메소드의 리턴 Issue

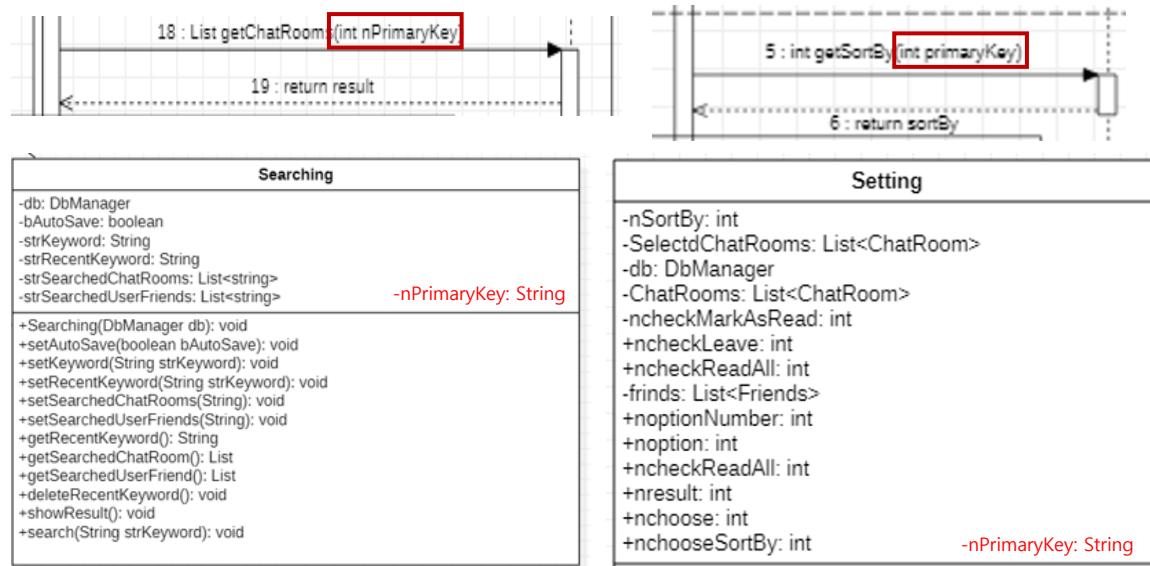


리턴값이 없는 메소드의  
리턴 명시가 있는 이슈

```
public int deleteData(String option, Object Data) {  
    int result=-1;  
    switch(option) {  
        case "Searching":  
            System.out.println("최근 검색어 삭제");  
            result = 0;  
            break;  
    }  
    return result;  
}
```

메소드의 선언을 반환값이  
있도록 해 주어 해결

## ● 구현에 있어 어려웠던 점 : 다이어그램간 싱크 Issue



```
class Searching{
    private DbManager db = new DbManager();
    private boolean bAutoSave = true; // 기본
    private String strKeyword;
    private ArrayList<String> strRecentKeyw
    private ArrayList<ChatRoom> SearchedChat
    private ArrayList<User> SearchedUserFrie
    private ArrayList list= new ArrayList();
    Scanner sc = new Scanner(System.in);
    int nPrimaryKey = 0;
}

class Setting{
    Scanner sc = new Scanner(System.in);
    private int nSortBy;
    private List<ChatRoom> SelectChatRooms;
    private DbManager db;
    private List<ChatRoom> ChatRooms;
    private int ncheckMarkAsRead;
    public int ncheckLeave;
    public int ncheckReadAll;
    public int noptionNumber;
    public int noption;
    public int nresult;
    public int nchoose;
    public int nchooseSortBy;
    public int nPrimaryKey;
    public int allVoteCounts;
}
```

시퀀스 다이어그램에는 쓰이는  
변수가 클래스 다이어그램에는  
없는 이슈

클래스에 새로운 변수를  
선언하여 해결

## ● 구현에 있어 어려웠던 점 : 다이어그램간 싱크 Issue

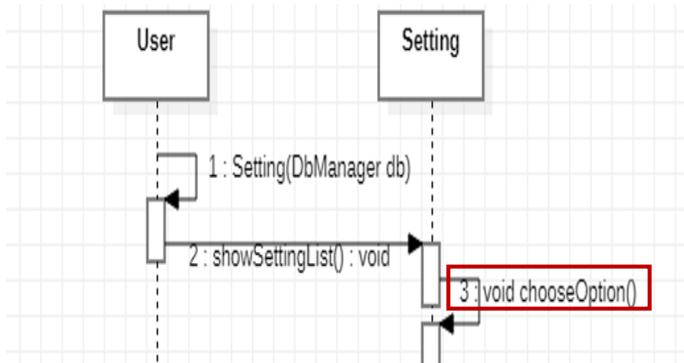


시퀀스 다이어그램과 클래스  
다이어그램의 메소드 이름이  
다른 이슈

```
public void EditChatRoom() { //추가  
    System.out.println("채팅방 편집");  
    int result = db.connectDB(); //편집(2)(3): 데이터베이스연결  
    if(result == -1) //db접근에 이상이 있을경우  
    {  
        showException("Error"); //편집(4): 데이터베이스 문제확인  
        System.exit(0); //ExceptionError 편집(5)  
    }  
    else //db접근에 이상이 없을경우  
    {  
        List chatRoomsList = db.getChatRooms(nPrimaryKey); // 편집(6)(7)  
        if(chatRoomsList == null) // null이면 예외  
        {  
            showException("Error"); //편집(8)  
            System.exit(0); //ExceptionError 편집(9)  
        }  
        showEdit(); //편집(10)  
    }  
}
```

표기 오류임을 인지하고 클래스  
다이어그램에 맞추어서  
구현하여 해결

## ● 구현에 있어 어려웠던 점 : 다이어그램간 싱크 Issue



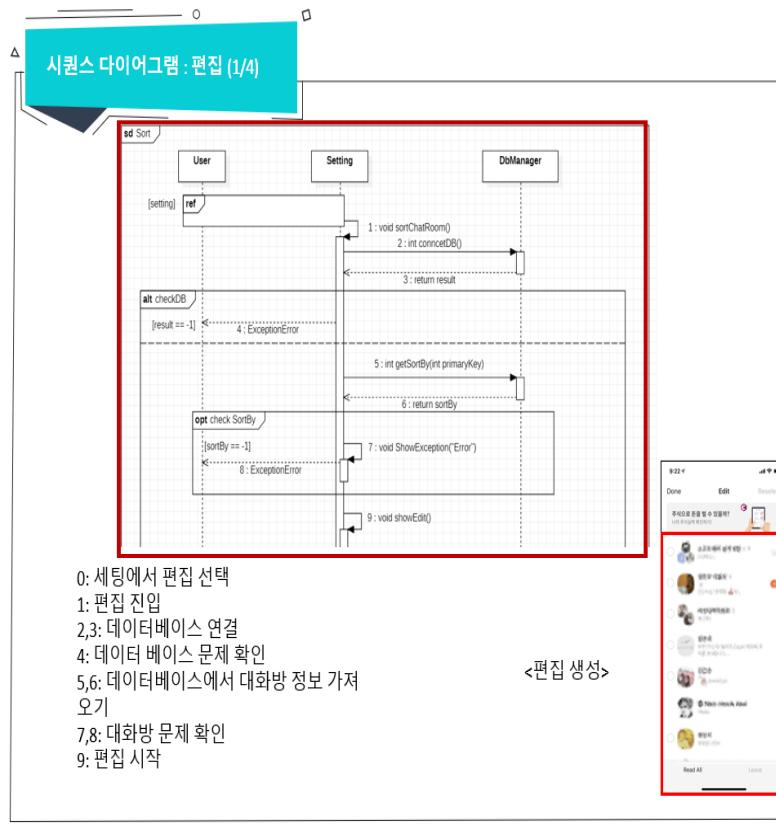
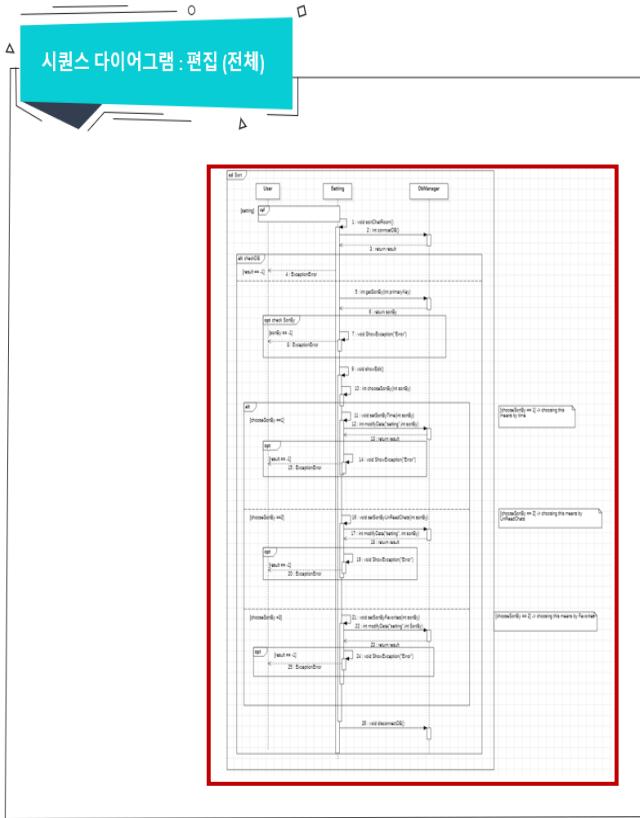
```
+chooseoption(): void  
+EditChatRoom(): void  
+showEdit(): void  
+chooseOption(): int  
+startReadAll(): void  
+checkReadAll(): int
```

시퀀스 다이어그램과 클래스  
다이어그램의 메소드 이름이  
다른 이슈

```
public void chooseoption() {  
    System.out.println("chooseOption");  
    System.out.print(">>");  
    int OptionNumber = sc.nextInt();
```

표기 오류임을 인지하고 클래스  
다이어그램에 맞추어서  
구현하여 해결

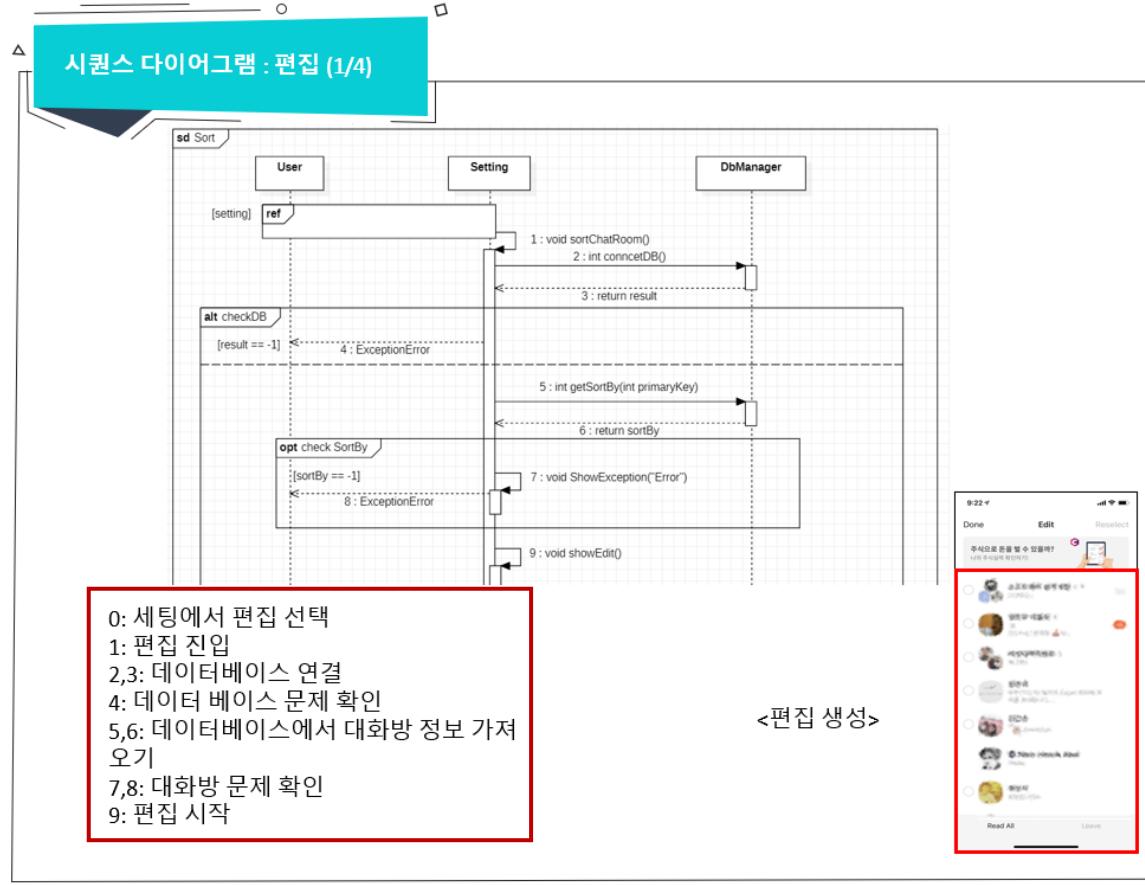
## ● 구현에 있어 어려웠던 점 : 다이어그램 설명 Issue



시퀀스 다이어그램 설명과  
싱크가 맞지 않는 그림  
(다른 시퀀스 다이어그램)  
첨부 이슈

해당 시퀀스 다이어그램  
원본파일을 받아 해결

## ● 구현에 있어 어려웠던 점 : 다이어그램 설명 Issue



시퀀스 다이어그램에  
누락된 설명이 있는 이슈

설계팀과의 소통을 통해 해결

## ● 구현에 있어 어려웠던 점 : 오타 Issue

[option == 2]

```
else if(option == 2)
{
    selectChatRooms(ChatRooms); //편집(19)
    int choose = chooseBetweenMarkAsReadAndLeave(); //편집(20)
    if(choose == 1)
    {
        startMarkAsRead(); //편집(21)
    }
}
```

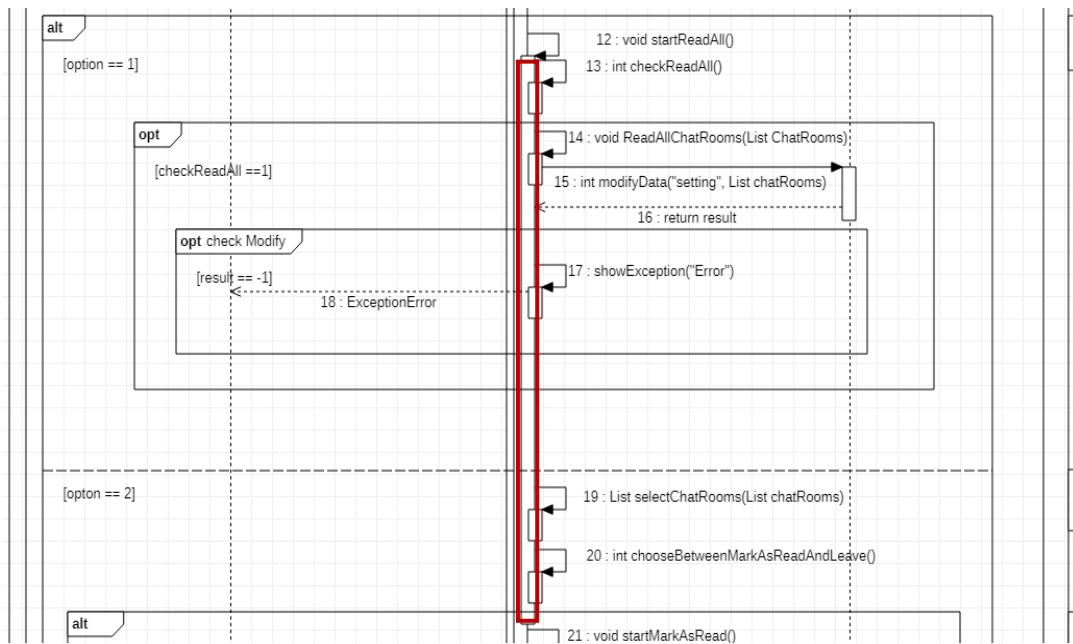
35: void disconnectDB()

```
db.disconnectDB(); //편집(35)
```

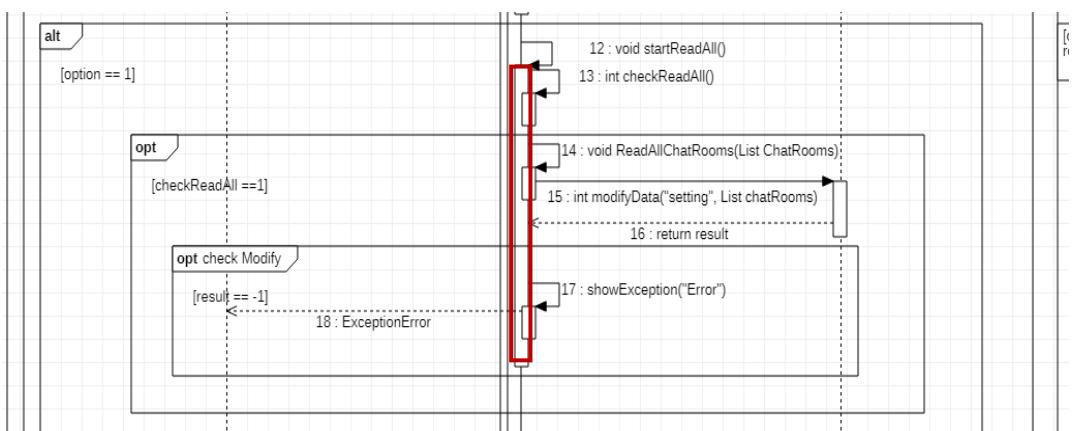
시퀀스 다이어그램에  
여러 오타 이슈

앞 뒤를 흐름을 보고 뜻하는  
변수나 메소드가 무엇인지  
인지하고 수정하여 해결

## ● 구현에 있어 어려웠던 점 : 엑티베이션 바 Issue

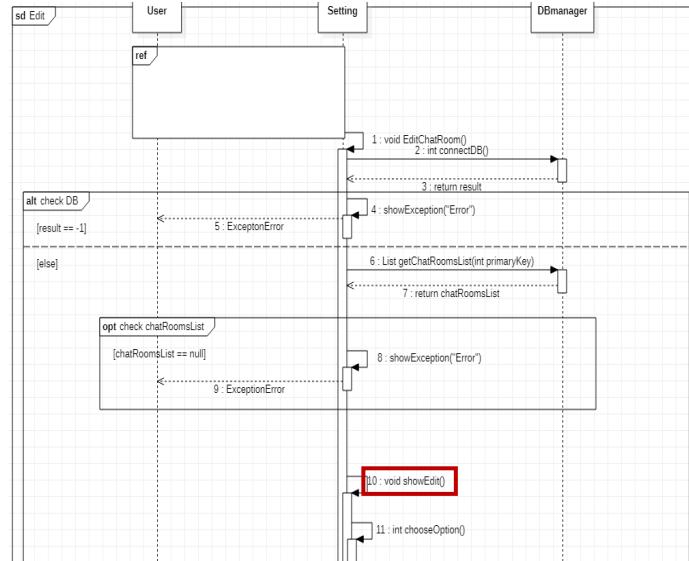


시퀀스 다이어그램의  
엑티베이션 바(12번) 길이가  
잘못 그려져 있는 이슈

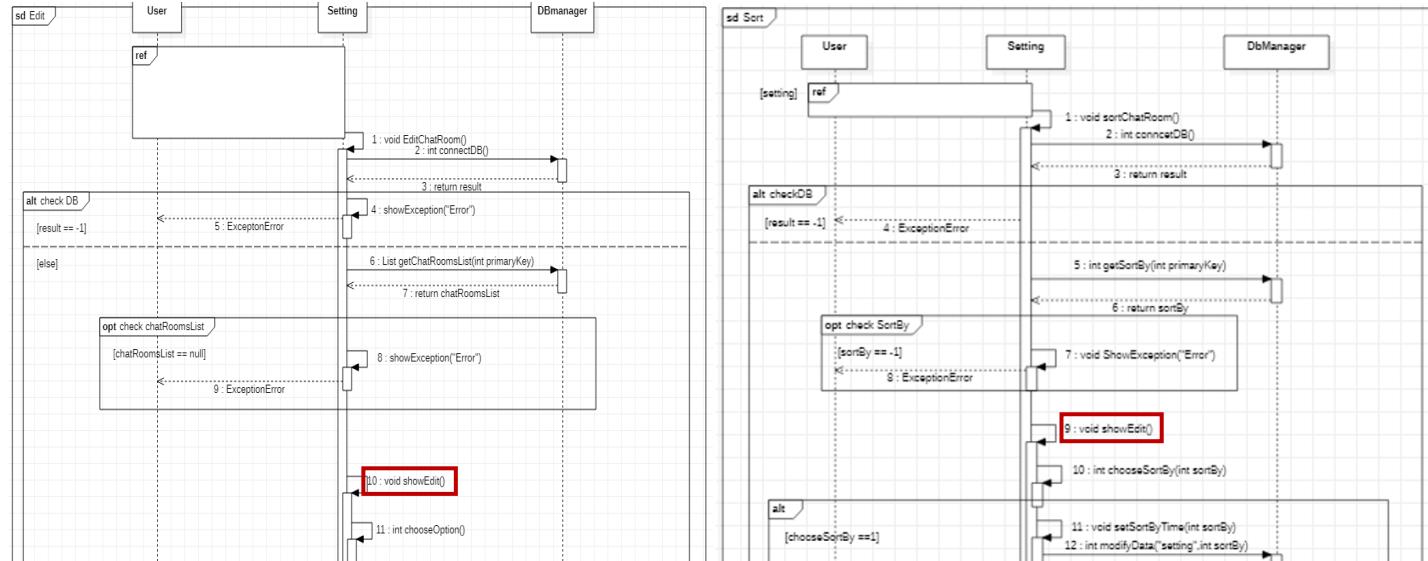


설계팀과의 소통을 통해  
엑티베이션 바를 수정하여 해결

## ● 구현에 있어 어려웠던 점 : 메소드 중복 Issue



```
public void showSort() {
    System.out.println("정렬기능을 보여준다");
    int chooseSortBy = chooseSortBy(nSortBy); //정렬(10)
    if(chooseSortBy == 1) //최근메시지 순
    {
        setSortByTime(nSortBy); //정렬(11)
    }
    else if(chooseSortBy == 2) //안읽은 메시지 순
    {
        setSortByUnReadChats(nSortBy); //정렬(16)
    }
    else if(chooseSortBy == 3)// 즐겨찾기순
    {
        setSortByFavorites(nSortBy); //정렬(21)
    }
}
```



다른 기능에 하나의 메소드를 사용하여 충돌하는 이슈

새로운 메소드를 생성하여 해결

---

# Q & A

---