# 카카오톡 친구 기능 구현

과목 : 소프트웨어 설계

조 : D조

조원 : 전유승, 권철현, 김종민, 최영은

# 목차

# 01

## 개요

# 01. 개요

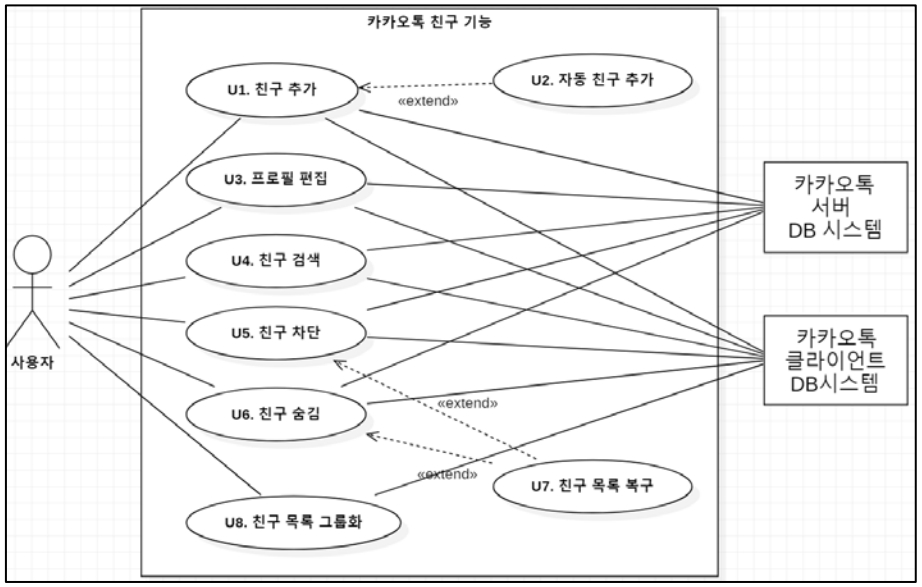
B조

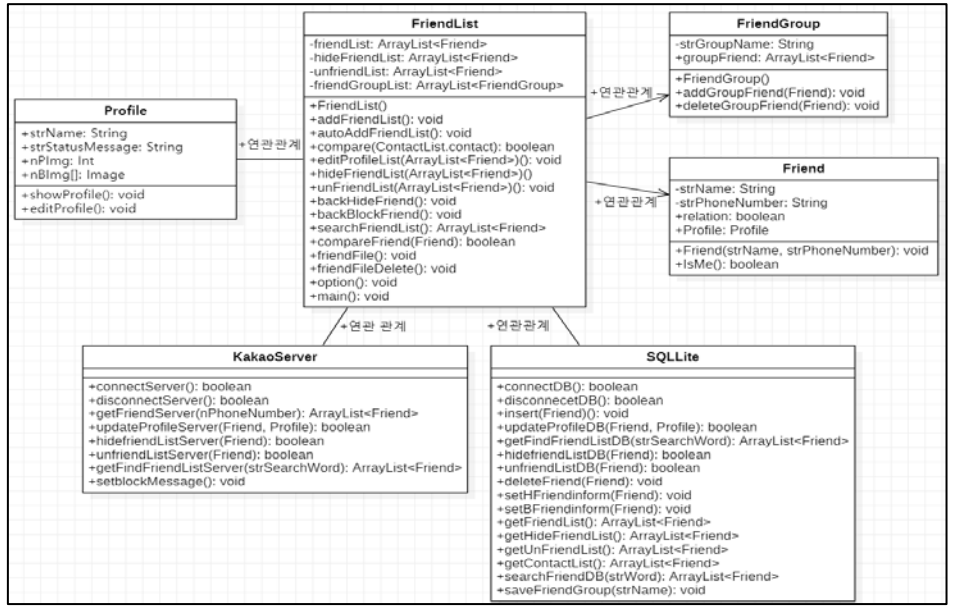
친구 기능 설계서

설계서를 바탕으로 구현

구현 언어는 JAVA, IDE는 IntelliJ 를 사용하였습니다.

# 01. 개요




8가지의 기능을 총 6개의 클래스로 구현

Friend.java

FriendGroup.java

FriendList.java

KaKaoServer.java

Profile.java

SQLLite.java

# 01. 개요


FriendList.java

대부분의 기능들은 해당 클래스에서 구현.
친구 추가, 자동 친구추가, 프로필 편집, 친구 검색, 친구 숨김, 친구 차단, 친구목록 복구, 친구목록 그룹화 기능을 가짐.


Friend.java

구조체 처럼 사용하는 클래스, 친구에 대한 정보를 가짐.
생성자와 getter, setter 를 가지고, IsMe 라는 정보가 본인인지 확인하는 메소드를 가짐.


FriendGroup.java

구조체 처럼 사용하는 클래스, ArrayList<Friend> 를 가짐.
생성자와 ArrayList에 추가, 제거하는 메소드를 가짐.

# 01. 개요


Profile.java

구조체처럼 사용하는 클래스, 프로필에 대한 정보를 가짐
showProfile 로 프로필을 보여주고, editProfile 로 프로필을 수정함.


KaKaoServer.java

카카오 서버에 연결하는 클래스, 서버에 연결해서 정보를 업데이트 하거나, 가져오는 메소드 등을 가짐.


SQLLite.java

데이터베이스에 연결하는 클래스, 데이터베이스에 연결해서 데이터를 가져오거나 보내는 메소드 등을 가짐.

# 02

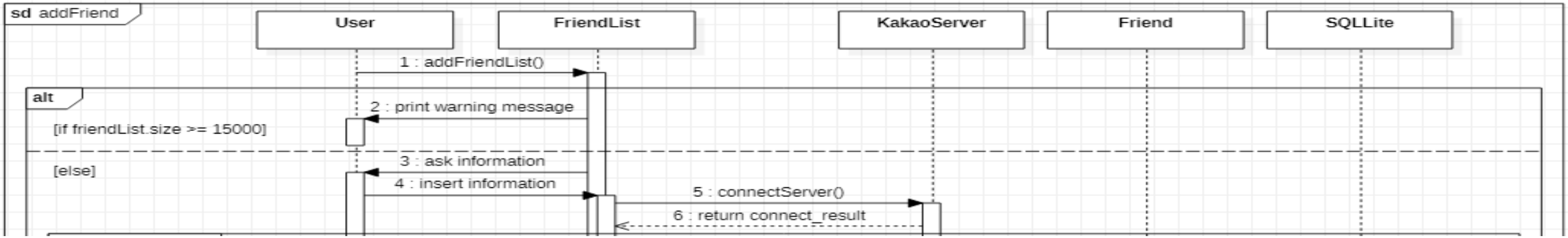
## 기능별 구현

# 02. 기능별 구현 – U1.친구추가 (0 / 4)


FriendList.java

```java
//---------------------------------- U1 친구 추가 ----------------------------------------------------
public void addFriendList() {                                   //U1.Sequence 1
    if(friendList.size() > 15000) {
        System.out.println("Error!");                          //U1.Sequence  2
    }
    else{
        System.out.println("Ask Information");                 //U1.Sequence 3

        String strName = sc.next();                            //U1.Sequence 4
        int nPhoneNumber = sc.nextInt();                       //U1.Sequence 4

        boolean connectResult = kaKaoServer.connectServer();  //U1.Sequence 5, 6
        if(connectResult == false) {
            System.out.println("Error!");                      //U1.Sequence 7
        }
        else{
            Friend returnFriend = kaKaoServer.getFriendServer(nPhoneNumber);    //U1.Sequence 8, 9
            if(returnFriend == null) {
                System.out.println("Error!");                  //U1.Sequence 10
            }
            else if(returnFriend.IsMe() == true) {
                System.out.println("Error!");                  //U1.Sequence 11
            }
            else{
                boolean connectDB_result = sqlLite.connectDB();     //U1.Sequence 12, 13

                if(connectDB_result == false){
                    System.out.println("Error!");              //U1.Sequence 14
                }
                else{
                    Friend friend = new Friend(strName, nPhoneNumber);   //U1.Sequence 15
                    /* U1.Sequence 16, 생성자에는 return 이 없다. */
                    boolean insert_result = sqlLite.Insert(friend);      //U1.Sequence 17, 18
                    if(insert_result == true) {
                        System.out.println("OK!");             //U1.Sequence 19
                    }
                    sqlLite.disconnectDB();                    //U1.Sequence 20
                }
            }
            kaKaoServer.disconnectServer();                    //U1.Sequence 21
        }
    }
}
```

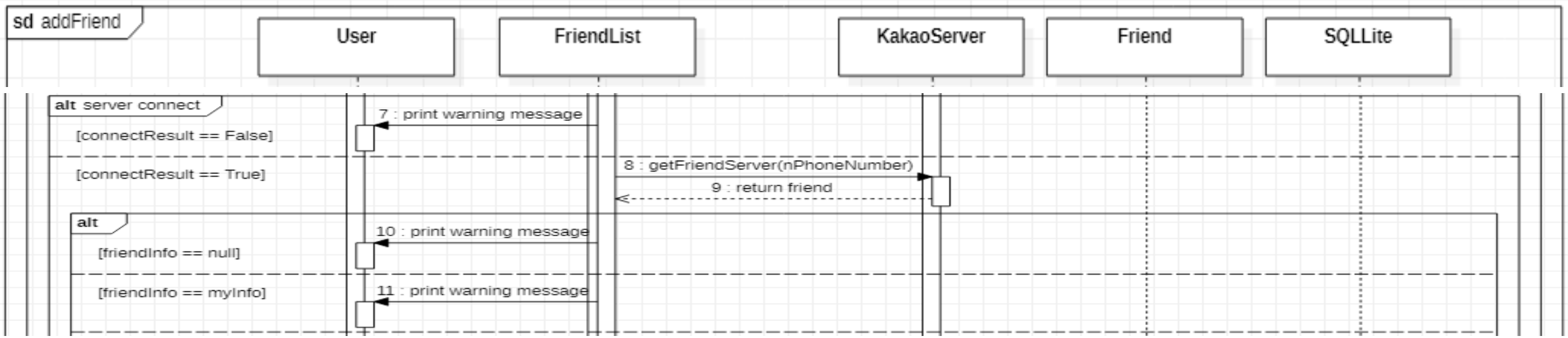
# 02. 기능별 구현 – U1.친구추가 (1 / 4)




FriendList.java

```java
//---------------------------------- U1 친구 추가 ----------------------------------
public void addFriendList() {                                   //U1.Sequence 1
    if(friendList.size() > 15000) {
        System.out.println("Error!");                          //U1.Sequence  2
    }
    else{
        System.out.println("Ask Information");                 //U1.Sequence 3

        String strName = sc.next();                            //U1.Sequence 4
        int nPhoneNumber = sc.nextInt();                       //U1.Sequence 4

        boolean connectResult = kaKaoServer.connectServer(); //U1.Sequence 5, 6
```

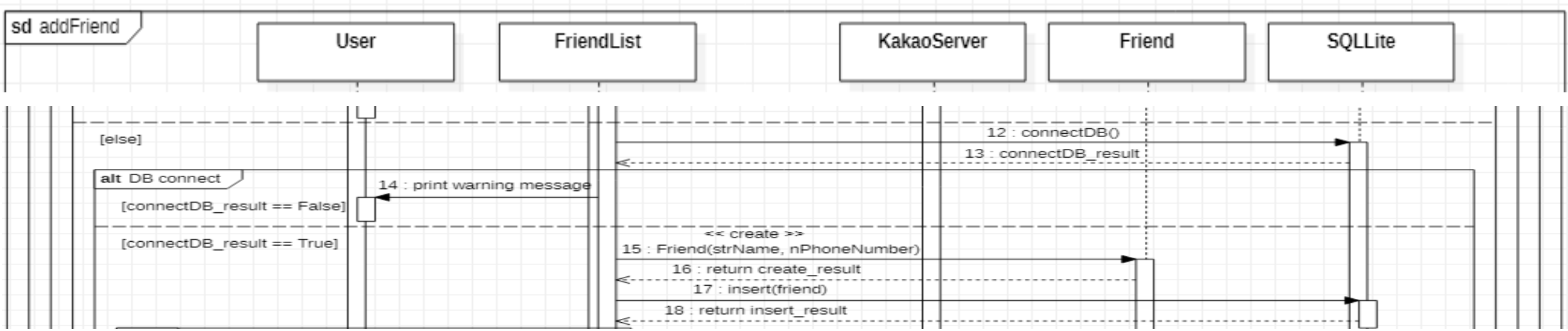
# 02. 기능별 구현 – U1.친구추가 (2 / 4)




FriendList.java

```
31          if(connectResult == false) {
32              System.out.println("Error!");                  //U1.Sequence 7
33          }
34          else{
35              Friend returnFriend = kaKaoServer.getFriendServer(nPhoneNumber);   //U1.Sequence 8, 9
36              if(returnFriend == null) {
37                  System.out.println("Error!");              //U1.Sequence 10
38              }
39              else if(returnFriend.IsMe() == true) {
40                  System.out.println("Error!");              //U1.Sequence 11
41              }
```

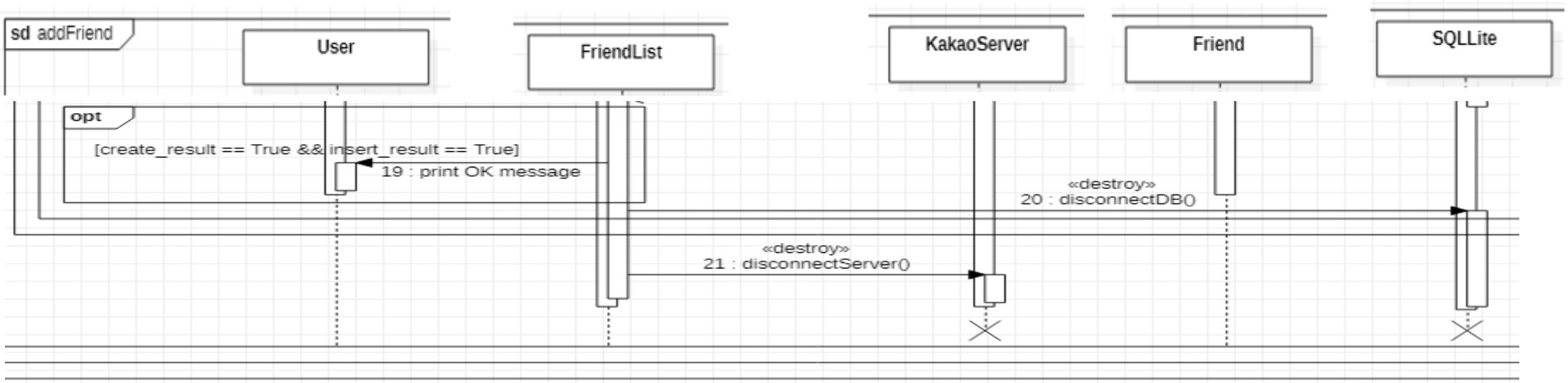
# 02. 기능별 구현 – U1.친구추가 (3 / 4)




FriendList.java

```
42              else{
43                  boolean connectDB_result = sqlLite.connectDB();    //U1.Sequence 12, 13
44
45                  if(connectDB_result == false){
46                      System.out.println("Error!");          //U1.Sequence 14
47                  }
48                  else{
49                      Friend friend = new Friend(strName, nPhoneNumber);  //U1.Sequence 15
50                      /* U1.Sequence 16, 생성자에는 return 이 없다. */
51                      boolean insert_result = sqlLite.Insert(friend);     //U1.Sequence 17, 18
```

# 02. 기능별 구현 – U1.친구추가 (4 / 4)




FriendList.java

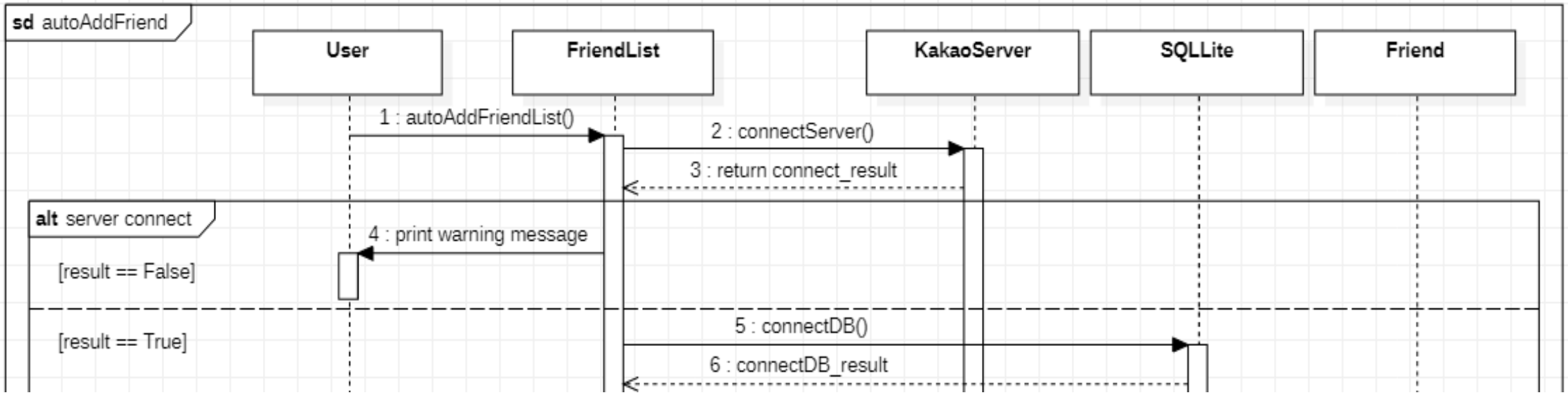
```java
52                      if(insert_result == true) {
53                          System.out.println("OK!");          //U1.Sequence 19
54                      }
55                      sqlLite.disconnectDB();                 //U1.Sequence 20
56                  }
57              }
58              kaKaoServer.disconnectServer();                 //U1.Sequence 21
59          }
60      }
61  }
```

# 02. 기능별 구현 – U2.자동친구추가 (0 / 4)


FriendList.java

```java
//---------------------------------- U2 자동 친구 추가 ----------------------------------------
public void autoAddFriends() {                              //U2.Sequence 1
    boolean result = kaKaoServer.connectServer();           //U2.Sequence 2, 3
    if(result == false) {
        System.out.println("Error!");                       //U2.Sequence 4
    }
    else{
        boolean connectDB_result = sqlLite.connectDB();     //U2.Sequence 5, 6
        if(connectDB_result == false){
            System.out.println("Error!");                   //U2.Sequence 7
        }
        else{
            ArrayList<Friend> contactList = sqlLite.getContactList();   //U2.Sequence 8, 9
            for(int i = 0; i<contactList.size(); i ++){
                boolean isInclude = compare(contactList.get(i));   //U2.Sequence 10, 11
                if(isInclude == false){
                    Friend friend = kaKaoServer.getFriendServer(contactList.get(i).getnPhoneNumber());
                    //U2.Sequence 12, 13, 14
                    /* U2.Sequence 15, 생성자는 리턴이 없음. */
                    boolean insert_result = sqlLite.Insert(friend);     //U2.Sequence 16, 17
                    if(insert_result == true){
                        System.out.println("OK!");          //U2.Sequence 18
                    }
                }
            }
            sqlLite.disconnectDB();                         //U2.Sequence 19
        }
        kaKaoServer.disconnectServer();                     //U2.Sequence 20
    }
}
/* 시퀀스 다이어그램: ContactList.contact, 하지만 Friend 클래스에는 contact 맴버 변수 없음 */
public boolean compare(ArrayList<Friend> contactList) {
    boolean isInclude = true;       //U2.Sequence 10
    return true;                    //U2.Sequence 11
}
```

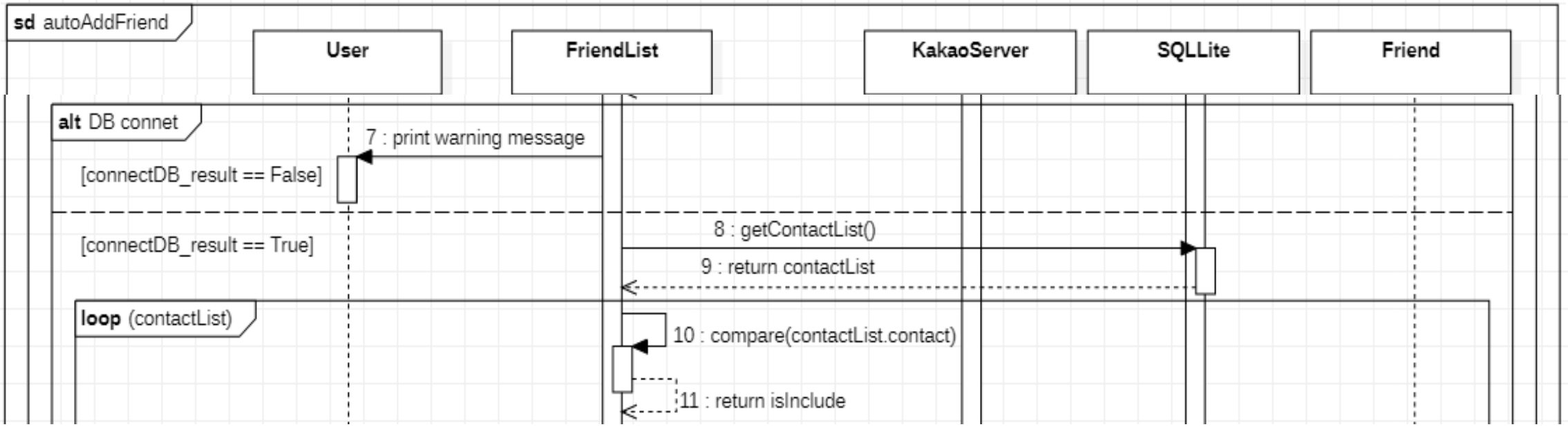
# 02. 기능별 구현 – U2.자동친구추가 (1 / 4)




FriendList.java

```java
//---------------------------------- U2 자동 친구 추가 ----------------------------
public void autoAddFriends() {                                  //U2.Sequence 1
    boolean result = kaKaoServer.connectServer();               //U2.Sequence 2, 3
    if(result == false) {
        System.out.println("Error!");                          //U2.Sequence 4
    }
    else{
        boolean connectDB_result = sqlLite.connectDB();         //U2.Sequence 5, 6
```

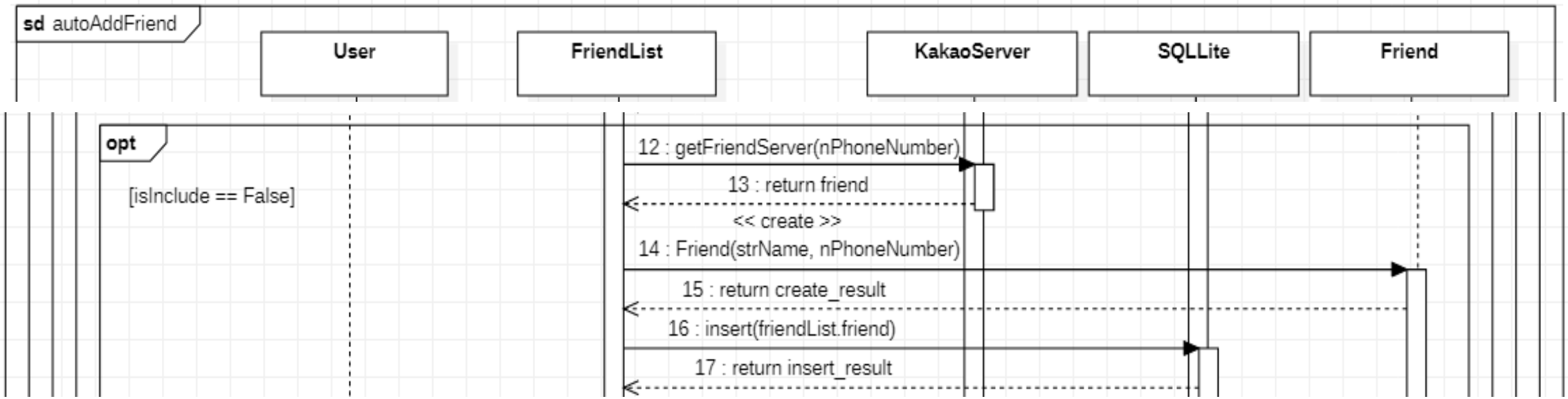
# 02. 기능별 구현 – U2.자동친구추가 (2 / 4)




FriendList.java

```java
72          if(connectDB_result == false){
73              System.out.println("Error!");                    //U2.Sequence 7
74          }
75          else{
76              ArrayList<Friend> contactList = sqlLite.getContactList();   //U2.Sequence 8, 9
77              for(int i = 0; i<contactList.size(); i ++){
78                  boolean isInclude = compare(contactList.get(i));   //U2.Sequence 10, 11
```

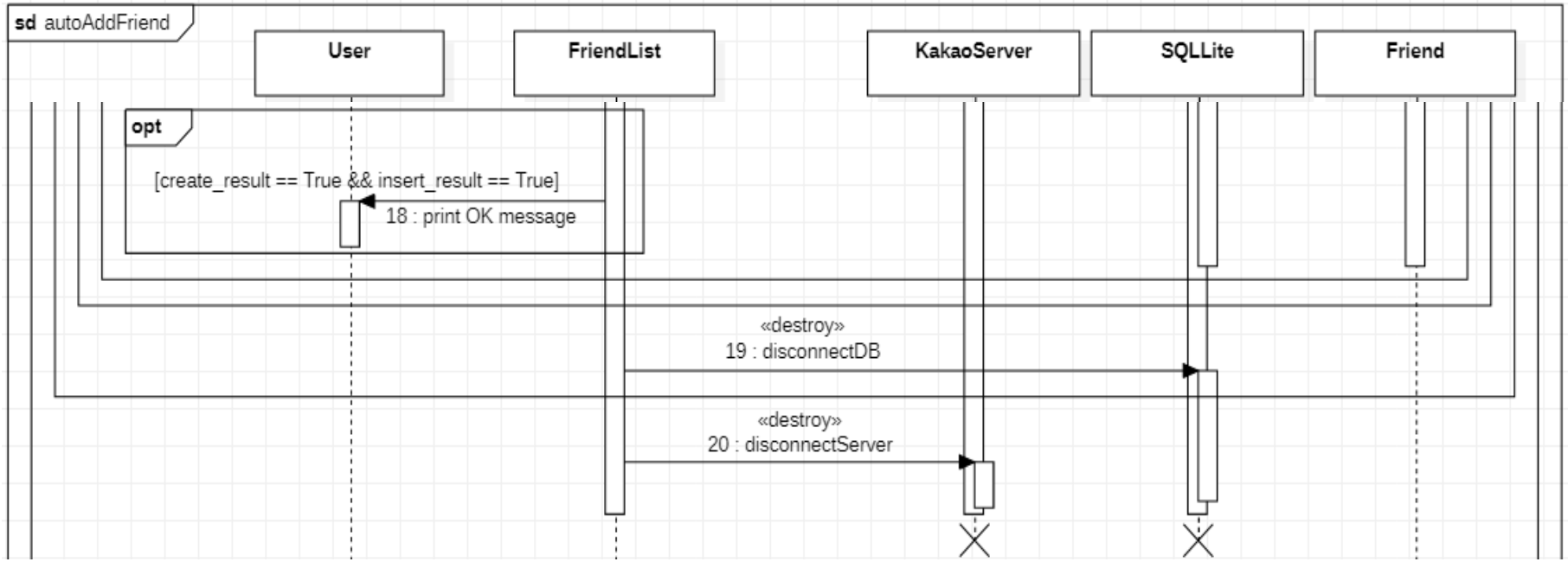
# 02. 기능별 구현 – U2.자동친구추가 (3 / 4)




FriendList.java

```
79              if(isInclude == false){
80                  Friend friend = kaKaoServer.getFriendServer(contactList.get(i).getnPhoneNumber());
81                  //U2.Sequence 12, 13, 14
82                  /* U2.Sequence 15, 생성자는 리턴이 없음. */
83                  boolean insert_result = sqlLite.Insert(friend);     //U2.Sequence 16, 17
```

# 02. 기능별 구현 – U2.자동친구추가 (4 / 4)




FriendList.java

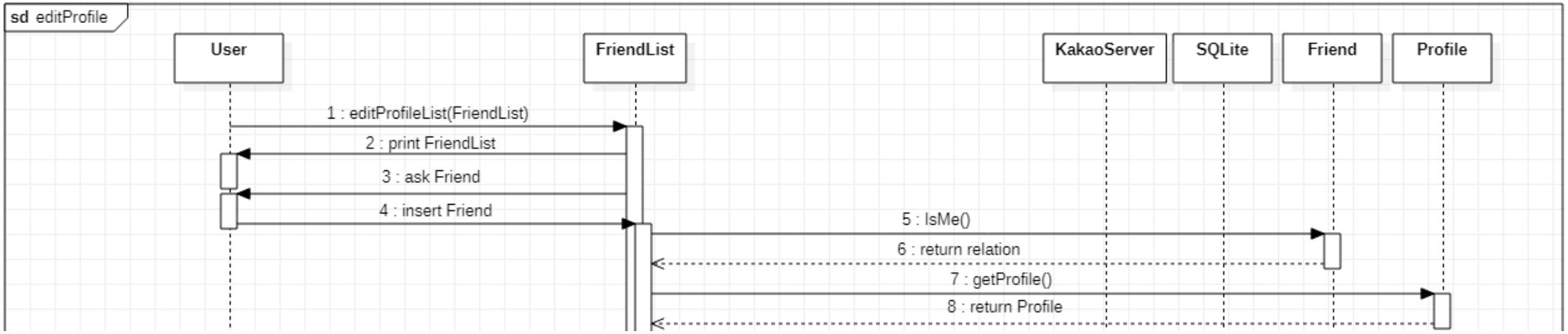
```
84                      if(insert_result == true){
85                          System.out.println("OK!");          //U2.Sequence 18
86                      }
87                  }
88              }
89              sqlLite.disconnectDB();                         //U2.Sequence 19
90          }
91          kaKaoServer.disconnectServer();                     //U2.Sequence 20
92      }
93  }
```

# 02. 기능별 구현 – U3.프로필 편집 (0 / 5)


FriendList.java

```java
public void editProfileList(ArrayList<Friend> friendList) { //U3.Sequence 1
    for(int i = 0; i < friendList.size(); i++) {
        System.out.println(friendList.get(i));              //U3.Sequence 2
    }
    System.out.println("Select Friend");                    //U3.Sequence 3
    int Insert_Friend = sc.nextInt();                       //U3.Sequence 4
    /* U3.4: Insert Friend 라는 메소드가 없는데, 활성화를 가짐 정정 필요 */
    boolean relation = friendList.get(Insert_Friend).IsMe();            //U3.Sequence 5, 6
    Profile profile = friendList.get(Insert_Friend).getProfile();       //U3.Sequence 7, 8
    if(relation == true){
        /* U3.9: print Profile 이라고 되어 있음. 메소드로 표현하는 것이 맞음. */
        profile.showProfile();                              //U3.Sequence 9
        System.out.println("Insert edit or exit");          //U3.Sequence 10
        String option = sc.next();                          //U3.Sequence 11
        if(option.equals("edit")) {
            System.out.println("Insert strName");           //U3.Sequence 12
            String strName = sc.next();                     //U3.Sequence 13
            System.out.println("Insert strStatusMessage");  //U3.Sequence 12
            String strStatusMessage = sc.next();            //U3.Sequence 13
            System.out.println("Insert nPlmg");             //U3.Sequence 12
            int nPlmg = sc.nextInt();                       //U3.Sequence 13
            System.out.println("Insert nBlmg");             //U3.Sequence 12
            int nBlmg = sc.nextInt();                       //U3.Sequence 13
            /* U3.13: Insert Friend 라는 메소드가 없는데, 활성화를 가짐 정정 필요 */

            /* U3.14: U1, U2 와 다르게 서버와 데이터베이스를 Create 로 생성 , 통일성 X */
            boolean connect_result = kaKaoServer.connectServer();   //U3.Sequence 14, 15
            if(connect_result == false){
                System.out.println("Error!");                   //U3.Sequence 16
            }
            else{
                boolean connectDB_result = sqlLite.connectDB();     //U3.Sequence 18, 19
                if(connectDB_result == false){
                    System.out.println("Error!");               //U3.Sequence 20
                }
                else{
                    profile.editProfile(strName, strStatusMessage, nPlmg, nBlmg);   //U3.Sequence 22
                    /* U3.23: 클래스 다이어그램에 editProfile 은 void 형식, 리턴 불가능 */
                    boolean insert_result1 = kaKaoServer.updateProfileServer(friendList.get(Insert_Friend), profile);   //U3.Sequence 24, 25
                    boolean insert_result2 = sqlLite.updateProfileDB(friendList.get(Insert_Friend), profile);       //U3.Sequence 26, 27
                    /* U3.28: 결과에 상관 없이 무조건 성공 메세지를 보냄. 아래와같이 if, else 로 수정함 */
                    if(insert_result1 == true && insert_result2 == true) {  //U3.Sequence 28
                        System.out.println("Success!");
                    }
                    else{
                        System.out.println("Error!");
                    }
                }
                sqlLite.disconnectDB();         //U3.Sequence 30
            }
            kaKaoServer.disconnectServer();     //U3.Sequence 29
        }
        else if(option.equals("exit")){
            return;                             //U3.Sequence 31
        }
    }
    else {
        profile.showProfile();                  //U3.Sequence 32
        System.out.println("Insert exit");      //U3.Sequence 33
        String option = sc.next();              //U3.Sequence 34
        if(option.equals("exit")) {
            return;                             //U3.Sequence 35
        }
    }
}
```

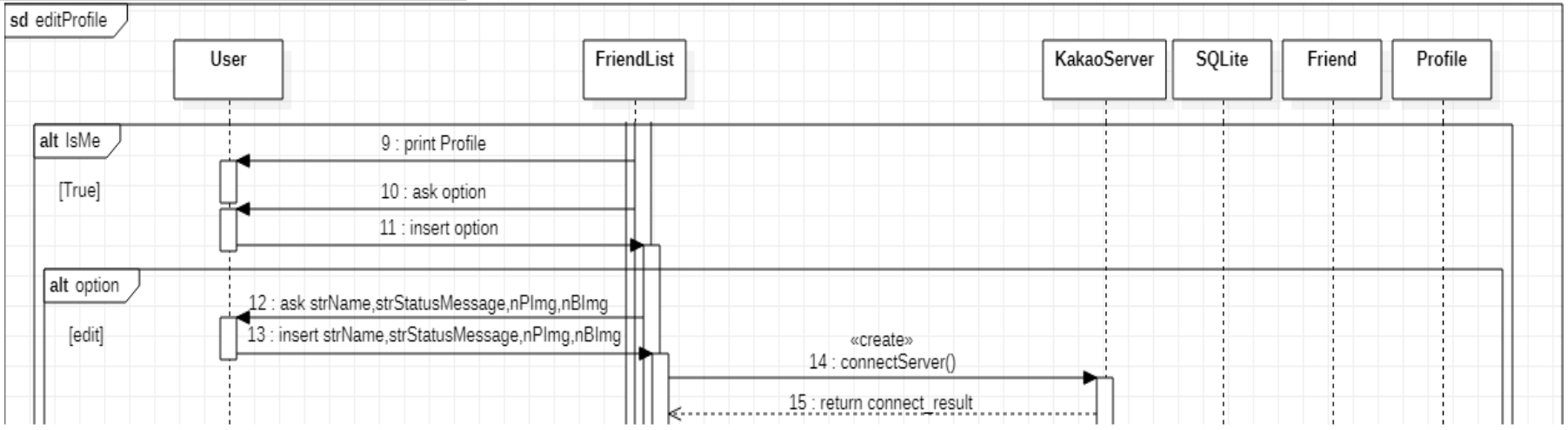
# 02. 기능별 구현 – U3.프로필 편집 (1 / 5)




FriendList.java

```java
public void editProfileList(ArrayList<Friend> friendList) { //U3.Sequence 1
    for(int i = 0; i < friendList.size(); i++) {
        System.out.println(friendList.get(i));                  //U3.Sequence 2
    }
    System.out.println("Select Friend");                        //U3.Sequence 3
    int Insert_Friend = sc.nextInt();                           //U3.Sequence 4
    /* U3.4: Insert Friend 라는 메소드가 없는데, 활성화를 가짐 정정 필요 */
    boolean relation = friendList.get(Insert_Friend).IsMe();            //U3.Sequence 5, 6
    Profile profile = friendList.get(Insert_Friend).getProfile();       //U3.Sequence 7, 8
```

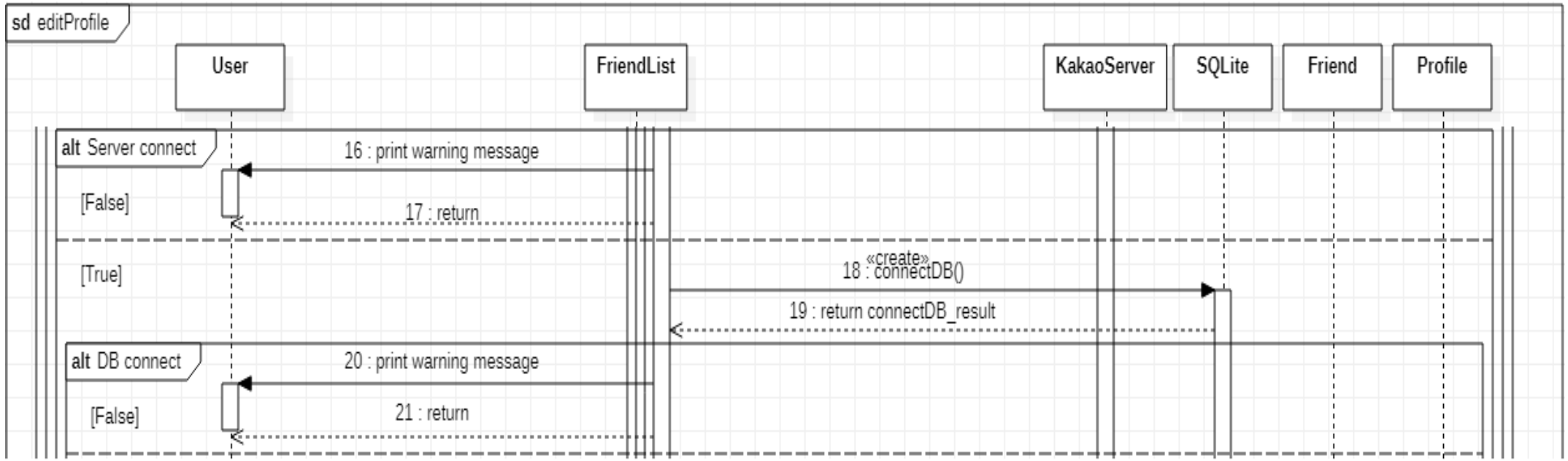
# 02. 기능별 구현 – U3.프로필 편집 (2 / 5)




FriendList.java

```
111         if(relation == true){
112             /* U3.9: print Profile 이라고 되어 있음. 메소드로 표현하는 것이 맞음. */
113             profile.showProfile();                                  //U3.Sequence 9
114             System.out.println("Insert edit or exit");              //U3.Sequence 10
115             String option = sc.next();                              //U3.Sequence 11
116             if(option.equals("edit")) {
117                 System.out.println("Insert strName");               //U3.Sequence 12
118                 String strName = sc.next();                         //U3.Sequence 13
119                 System.out.println("Insert strStatusMessage");      //U3.Sequence 12
120                 String strStatusMessage = sc.next();                //U3.Sequence 13
121                 System.out.println("Insert nPImg");                 //U3.Sequence 12
122                 int nPImg = sc.nextInt();                           //U3.Sequence 13
123                 System.out.println("Insert nBImg");                 //U3.Sequence 12
124                 int nBImg = sc.nextInt();                           //U3.Sequence 13
125                 /* U3.13: Insert Friend 라는 메소드가 없는데, 활성화를 가짐 정정 필요 */
126
127                 /* U3.14: U1, U2 와 다르게 서버와 데이터베이스를 Create 로 생성 , 통일성 X */
128                 boolean connect_result = kaKaoServer.connectServer();   //U3.Sequence 14, 15
```

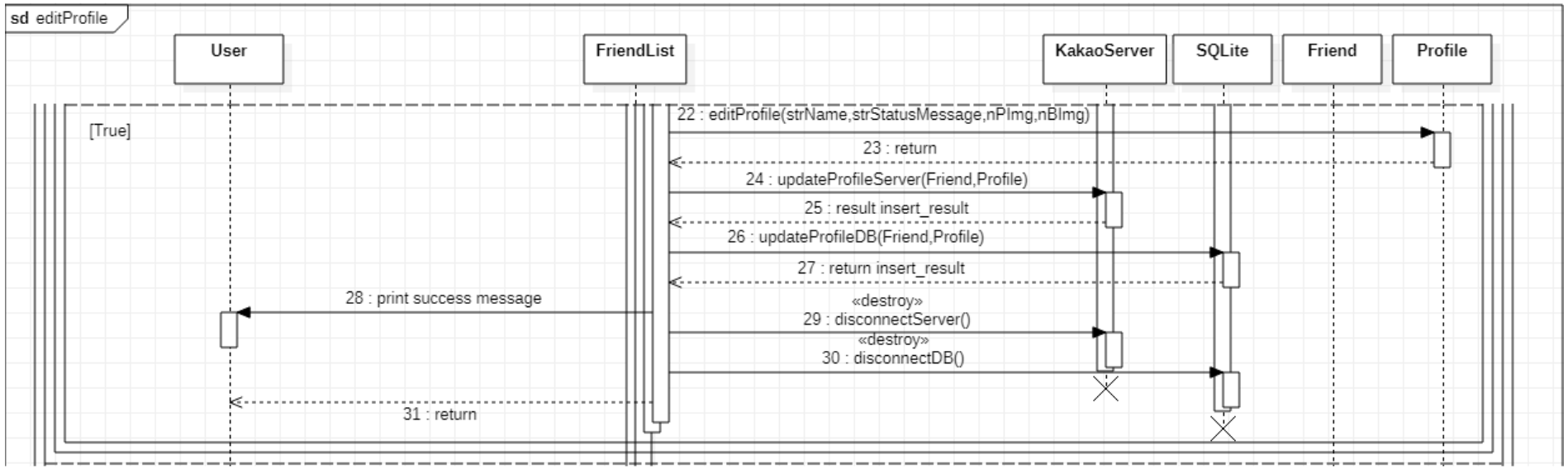
# 02. 기능별 구현 – U3.프로필 편집 (3 / 5)




FriendList.java

```java
129             if(connect_result == false){
130                 System.out.println("Error!");                    //U3.Sequence 16
131             }
132             else{
133                 boolean connectDB_result = sqlLite.connectDB();    //U3.Sequence 18, 19
134                 if(connectDB_result == false){
135                     System.out.println("Error!");                //U3.Sequence 20
136                 }
```

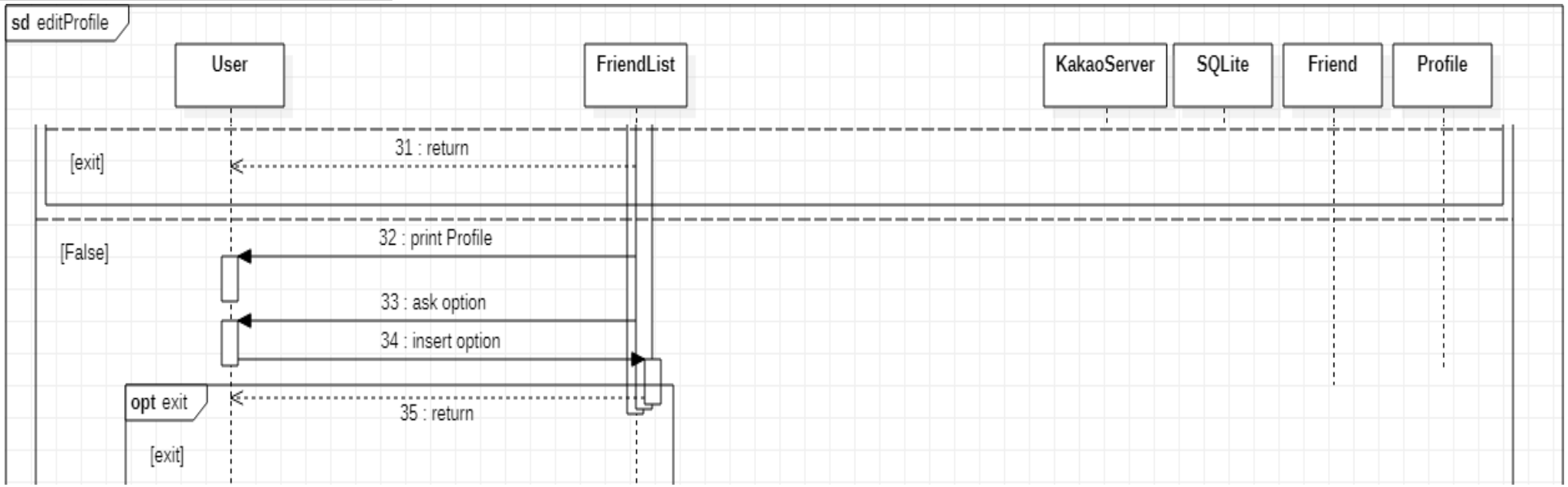
# 02. 기능별 구현 – U3.프로필 편집 (4 / 5)




FriendList.java

```
                else{
                    profile.editProfile(strName, strStatusMessage, nPImg, nBImg);    //U3.Sequence 22
                    /* U3.23: 클래스 다이어그램에 editProfile 은 void 형식, 리턴 불가능 */
                    boolean insert_result1 = kaKaoServer.updateProfileServer(friendList.get(Insert_Friend), profile);    //U3.Sequence 24, 25
                    boolean insert_result2 = sqlLite.updateProfileDB(friendList.get(Insert_Friend), profile);       //U3.Sequence 26, 27
                    /* U3.28: 결과에 상관 없이 무조건 성공 메세지를 보냄. 아래와같이 if, else 로 수정함 */
                    if(insert_result1 == true && insert_result2 == true) {   //U3.Sequence 28
                        System.out.println("Success!");
                    }
                    else{
                        System.out.println("Error!");
                    }
                }
                sqlLite.disconnectDB();          //U3.Sequence 30
            }
            kaKaoServer.disconnectServer();      //U3.Sequence 29
        }
```

# 02. 기능별 구현 – U3.프로필 편집 (5 / 5)




FriendList.java

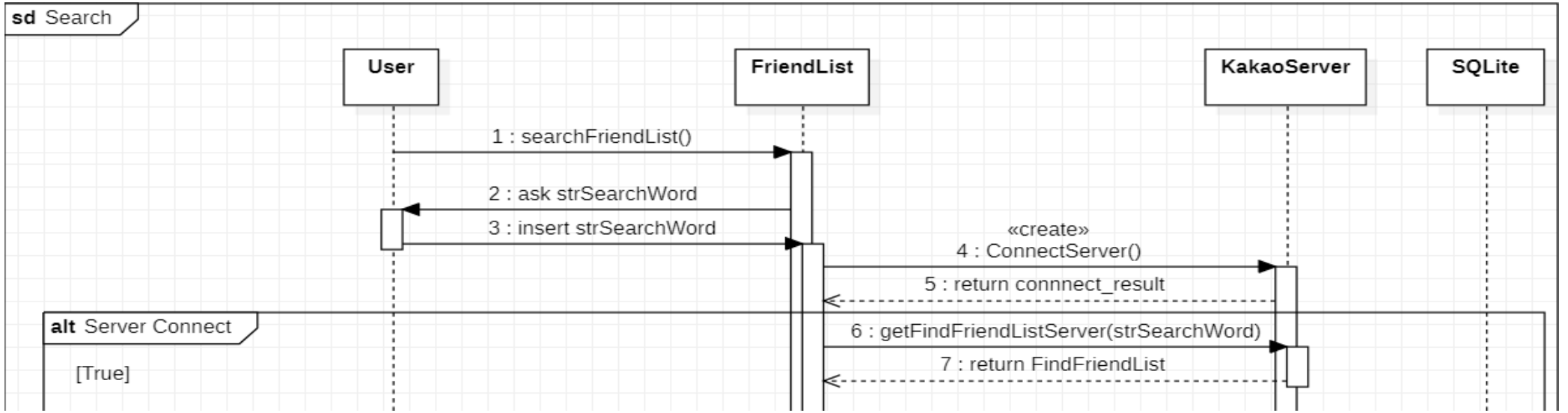
```java
            else if(option.equals("exit")){
                return;                                 //U3.Sequence 31
            }
        }
        else {
            profile.showProfile();                      //U3.Sequence 32
            System.out.println("Insert exit");          //U3.Sequence 33
            String option = sc.next();                  //U3.Sequence 34
            if(option.equals("exit")) {
                return;                                 //U3.Sequence 35
            }
        }
    }
```

# 02. 기능별 구현 – U4.친구 검색(0 / 4)


FriendList.java

```java
//---------------------------------- U4 친구 검색 ----------------------------------------------------
public ArrayList<Friend> searchFriendList() {        //U4.Sequence 1
    /* U4.Sequence 6, 7, 14, 15, 19를 위해 선언 필요 */
    ArrayList<Friend> FindFriendList = new ArrayList◇();

    System.out.println("Insert strSearchWorld");     //U4.Sequence 2
    String strSearchWorld = sc.next();               //U4.Sequence 3
    /* U4.Sequence 3에서 Insert strSearchWorld 라는 부분은 메소드가 없는데 활성화를 가짐 수정 필요 */

    boolean connect_result = kaKaoServer.connectServer(); //U4.Sequence 4, 5
    /* U4.4: U1, U2 와 다르게 서버와 데이터베이스를 Create 로 생성 , 통일성 X */
    if(connect_result == true){
        FindFriendList = kaKaoServer.getFindFriendListServer(strSearchWorld); //U4.Sequence 6, 7
        if(FindFriendList.size() > 0){
            for(int i = 0; i<FindFriendList.size(); i++) {
                System.out.println(FindFriendList.get(i));   //U4.Sequence 8
            }
        }
        else{
            System.out.println("No Friends");               //U4.Sequence 9
        }
        kaKaoServer.disconnectServer();                     //U4.Sequence 10
    }
    else{
        boolean connectDB_result = sqlLite.connectDB();     //U4.Sequence 11, 12
        if(connectDB_result == false) {
            System.out.println("Error!");                   //U4.Sequence 13
        }
        else{
            FindFriendList = sqlLite.getFindFriendListDB(strSearchWorld);    //U4.Sequence 14, 15
            if(FindFriendList.size() > 0){
                for(int i = 0; i<FindFriendList.size(); i++) {
                    System.out.println(FindFriendList.get(i));              //U4.Sequence 16
                }
            }
            else{
                System.out.println("No Friends");    //U4.Sequence 17
            }
            sqlLite.disconnectDB();                  //U4.Sequence 18
        }
    }
    return FindFriendList;                           //U4.Sequence 19
}
```

# 02. 기능별 구현 – U4.친구 검색(1 / 4)

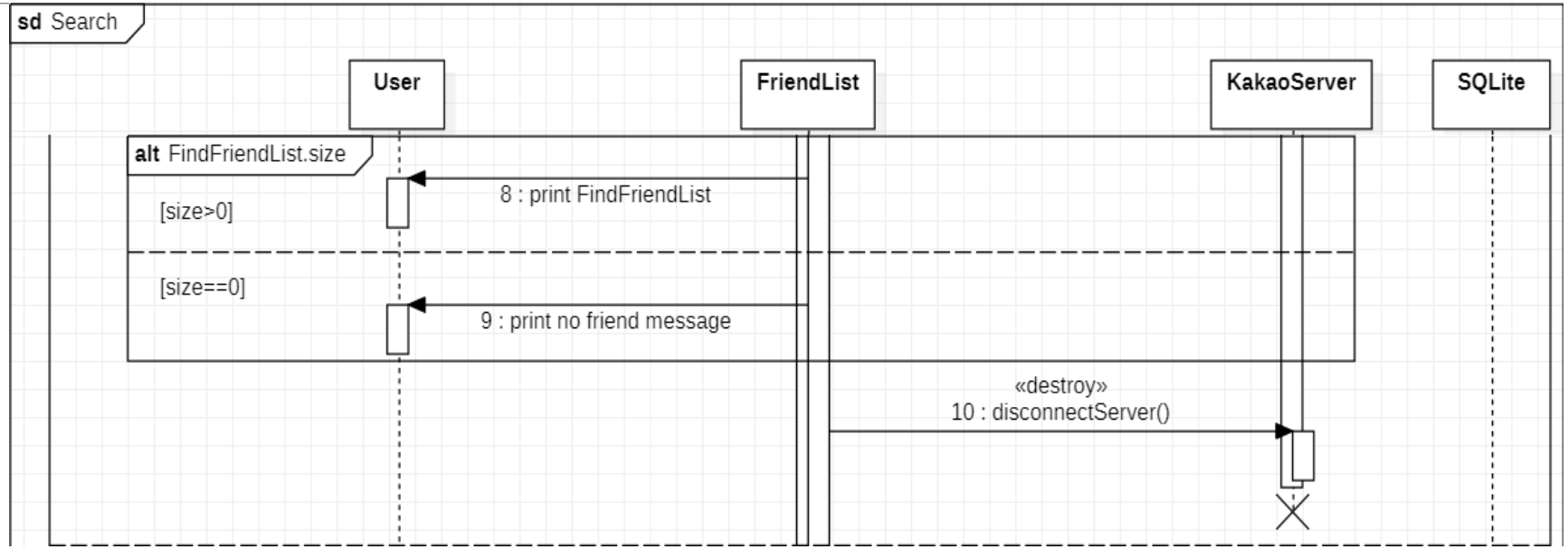



JAVA

FriendList.java

```java
//---------------------------------- U4 친구 검색 ----------------------------------------
public ArrayList<Friend> searchFriendList() {        //U4.Sequence 1
    /* U4.Sequence 6, 7, 14, 15, 19를 위해 선언 필요 */
    ArrayList<Friend> FindFriendList = new ArrayList<>();

    System.out.println("Insert strSearchWorld");     //U4.Sequence 2
    String strSearchWorld = sc.next();               //U4.Sequence 3
    /* U4.Sequence 3에서 Insert strSearchWorld 라는 부분은 메소드가 없는데 활성화를 가짐 수정 필요 */

    boolean connect_result = kaKaoServer.connectServer(); //U4.Sequence 4, 5
    /* U4.4: U1, U2 와 다르게 서버와 데이터베이스를 Create 로 생성 , 통일성 X */
    if(connect_result == true){
        FindFriendList = kaKaoServer.getFindFriendListServer(strSearchWorld); //U4.Sequence 6, 7
```

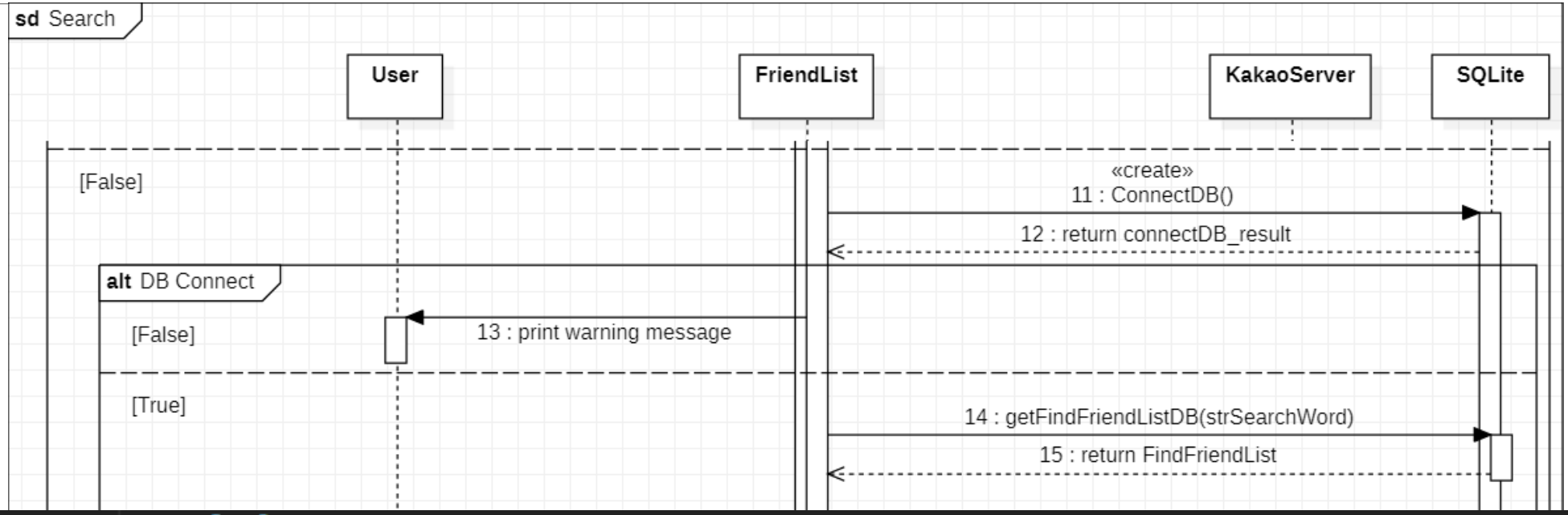
# 02. 기능별 구현 – U4.친구 검색(2 / 4)




FriendList.java

```
182         if(FindFriendList.size() > 0){
183             for(int i = 0; i<FindFriendList.size(); i++) {
184                 System.out.println(FindFriendList.get(i));   //U4.Sequence 8
185             }
186         }
187         else{
188             System.out.println("No Friends");               //U4.Sequence 9
189         }
190         kaKaoServer.disconnectServer();                     //U4.Sequence 10
191     }
```

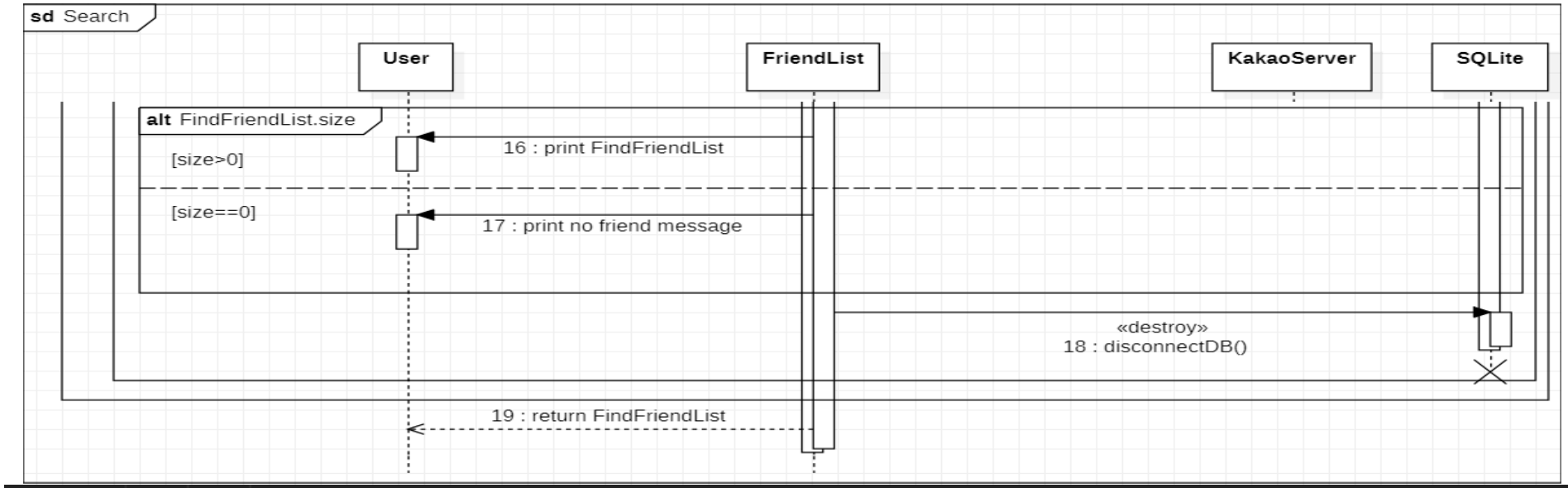
# 02. 기능별 구현 – U4.친구 검색(3 / 4)




FriendList.java

```
192        else{
193            boolean connectDB_result = sqlLite.connectDB();     //U4.Sequence 11, 12
194            if(connectDB_result == false) {
195                System.out.println("Error!");                   //U4.Sequence 13
196            }
197            else{
198                FindFriendList = sqlLite.getFindFriendListDB(strSearchWorld);    //U4.Sequence 14, 15
```

# 02. 기능별 구현 – U4.친구 검색(4 / 4)




FriendList.java

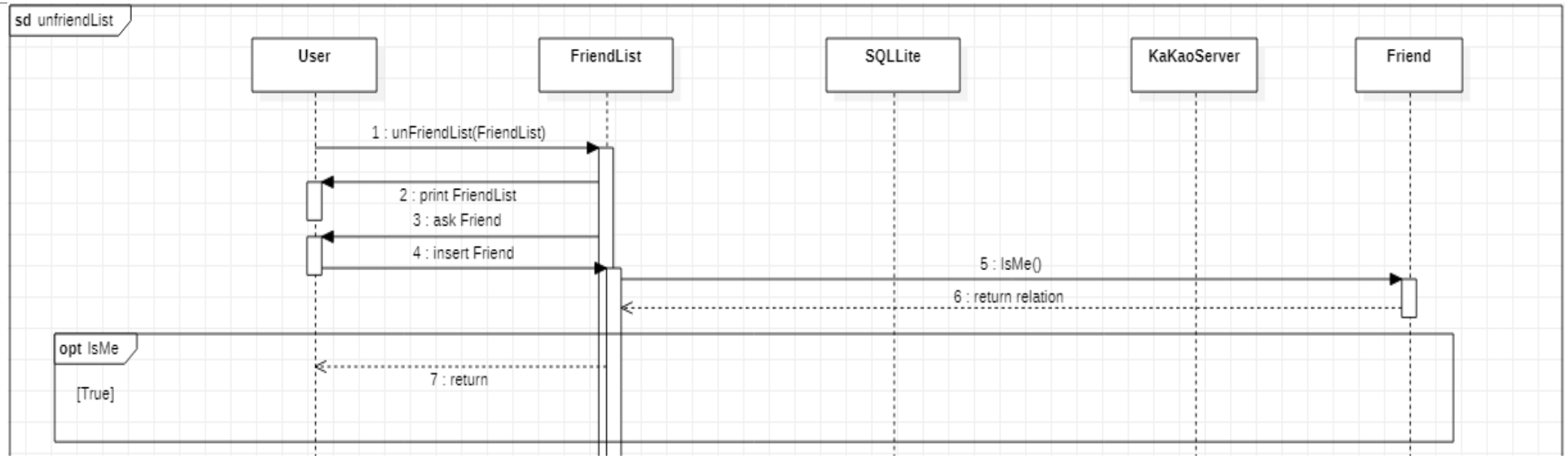
```java
199                 if(FindFriendList.size() > 0){
200                     for(int i = 0; i<FindFriendList.size(); i++) {
201                         System.out.println(FindFriendList.get(i));          //U4.Sequence 16
202                     }
203                 }
204                 else{
205                     System.out.println("No Friends");    //U4.Sequence 17
206                 }
207                 sqlLite.disconnectDB();                 //U4.Sequence 18
208             }
209         }
210         return FindFriendList;                          //U4.Sequence 19
211     }
```

# 02. 기능별 구현 – U5.친구 차단(0 / 4)


FriendList.java

```java
public void unFriendList(ArrayList<Friend> FriendList) {    //U5.Sequence 1
    for(int i = 0; i<FriendList.size(); i++){
        System.out.println(FriendList.get(i));  //U5.Sequence 2
    }
    System.out.println("Insert Friend");        //U5.Sequence 3
    int friend = sc.nextInt();                  //U5.Sequence 4
    /* U5.4: InsertFriend 라는 메소드는 없는데 활성화를 가짐. */

    boolean relation = FriendList.get(friend).IsMe();   //U5.Sequence 5, 6

    if(relation == true){
        return;                         //U5.Sequence 7
    }
    System.out.println("Insert unfriendOption : Cancel, Agree"); //U5.Sequence 8
    String unfriendOption = sc.next();                           //U5.Sequence 9
    /* U5.9: insert unfriendOption 라는 메소드는 없는데 활성화를 가짐. */

    /* U5 시퀀스 다이어그램 전체를 나타내는 다이어그램과, 세부 설명이 되어있는 부분이 다름. 일관성 X */
    /* 세부 설명이 되어있는 부분에 맞춰서 구현함 */
    if(unfriendOption.equals("Cancel")){
        return;                         //U5.Sequence 10
    }
    else if(unfriendOption.equals("Agree")){
        boolean connect_result = kaKaoServer.connectServer();    //U5.Sequence 11, 12
        if(connect_result == false){
            System.out.println("Error!");   //U5.Sequence 13
            return;
        }
        else{
            /* U5.15: U1, U2 와 다르게 서버와 데이터베이스를 Create 로 생성 , 통일성 X */
            boolean connectDB_result = sqlLite.connectDB();
            if(connectDB_result == false){
                System.out.println("Error!");   //U5.Sequence 17
                /* U5.18: 해당 부분을 실행하면 이후 26, 27 실행 불가, 따라서 삭제 필요 */
                /* return; */
            }
            else{
                //U5.Sequence 19, 21
                boolean insert_result1 = kaKaoServer.unFriendListServer(FriendList.get(friend));
                /* U5.21: setBlockMessage() 메소드의 리턴 값은 void 이지만, insert_result1 을 리턴함. */

                //U5.Sequence 22, 23
                boolean insert_result2 = sqlLite.unFriendListDB(FriendList.get(friend));
                /* U5.24: 결과에 상관 없이 무조건 성공 메세지를 보냄. 아래와같이 if, else 로 수정함 */
                if(insert_result1 == true && insert_result2 == true){
                    System.out.println("Success!");      //U5.Sequence24
                    /* U5.25: 해당 부분을 실행하면 이후 26, 27 실행 불가, 따라서 삭제 필요 */
                    /* return; */
                }
                sqlLite.disconnectDB();     //U5.Sequence 26
            }
            kaKaoServer.disconnectServer(); //U5.Sequence 27
        }
    }
}
```

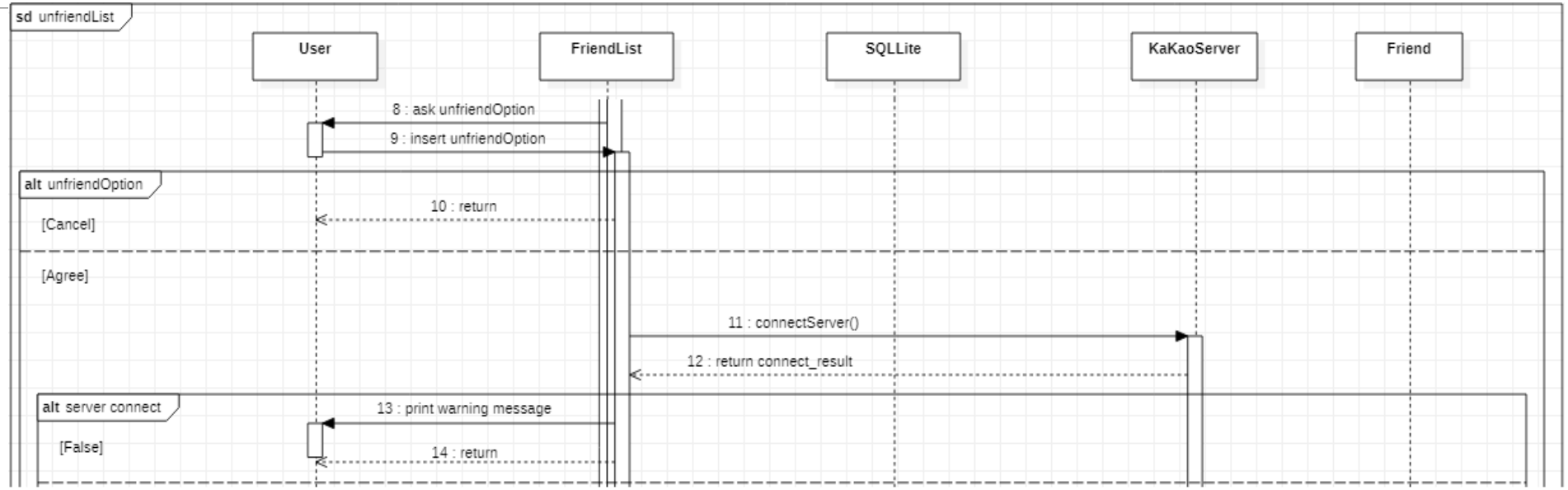
# 02. 기능별 구현 – U5.친구 차단(1 / 4)




FriendList.java

```java
public void unFriendList(ArrayList<Friend> FriendList) {    //U5.Sequence 1
    for(int i = 0; i<FriendList.size(); i++){
        System.out.println(FriendList.get(i));   //U5.Sequence 2
    }
    System.out.println("Insert Friend");          //U5.Sequence 3
    int friend = sc.nextInt();                    //U5.Sequence 4
    /* U5.4: InsertFriend 라는 메소드는 없는데 활성화를 가짐. */

    boolean relation = FriendList.get(friend).IsMe();    //U5.Sequence 5, 6

    if(relation == true){
        return;                          //U5.Sequence 7
    }
```

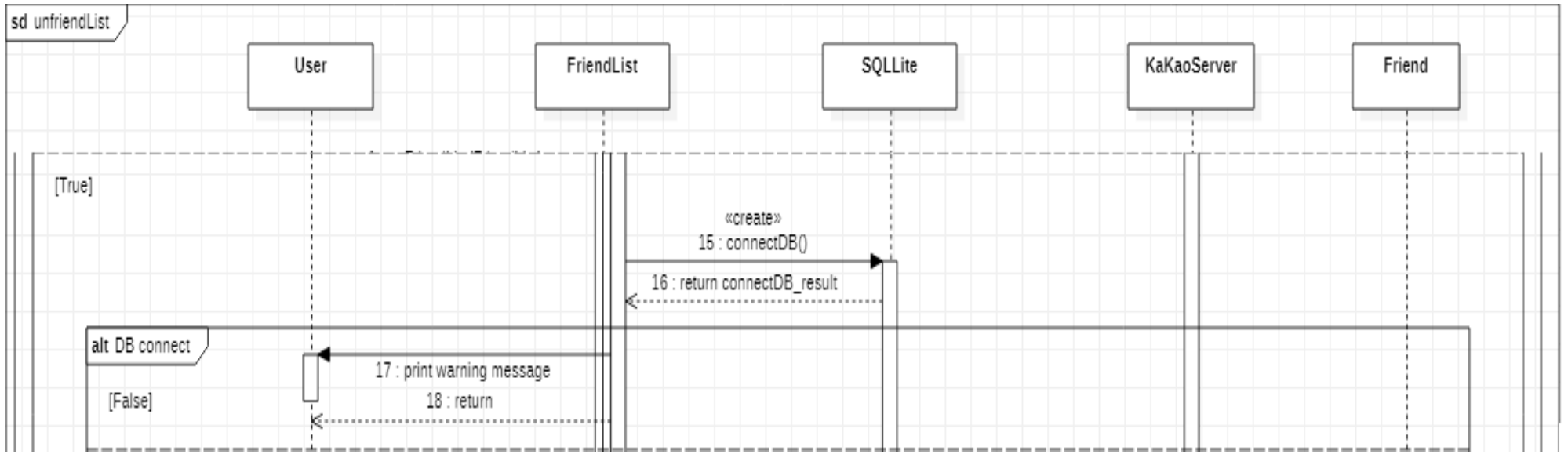
# 02. 기능별 구현 – U5.친구 차단(2 / 4)




FriendList.java

```
229         System.out.println("Insert unfriendOption : Cancel, Agree");//U5.Sequence 8
230         String unfriendOption = sc.next();                          //U5.Sequence 9
231         /* U5.9: insert unfriendOption 라는 메소드는 없는데 활성화를 가짐. */
232
233         /* U5 시퀀스 다이어그램 전체를 나타내는 다이어그램과, 세부 설명이 되어있는 부분이 다름. 일관성 X */
234         /* 세부 설명이 되어있는 부분에 맞춰서 구현함 */
235         if(unfriendOption.equals("Cancel")){
236             return;                         //U5.Sequence 10
237         }
238         else if(unfriendOption.equals("Agree")){
239             boolean connect_result = kaKaoServer.connectServer();     //U5.Sequence 11, 12
240             if(connect_result == false){
241                 System.out.println("Error!");    //U5.Sequence 13
242                 return;
243             }
```

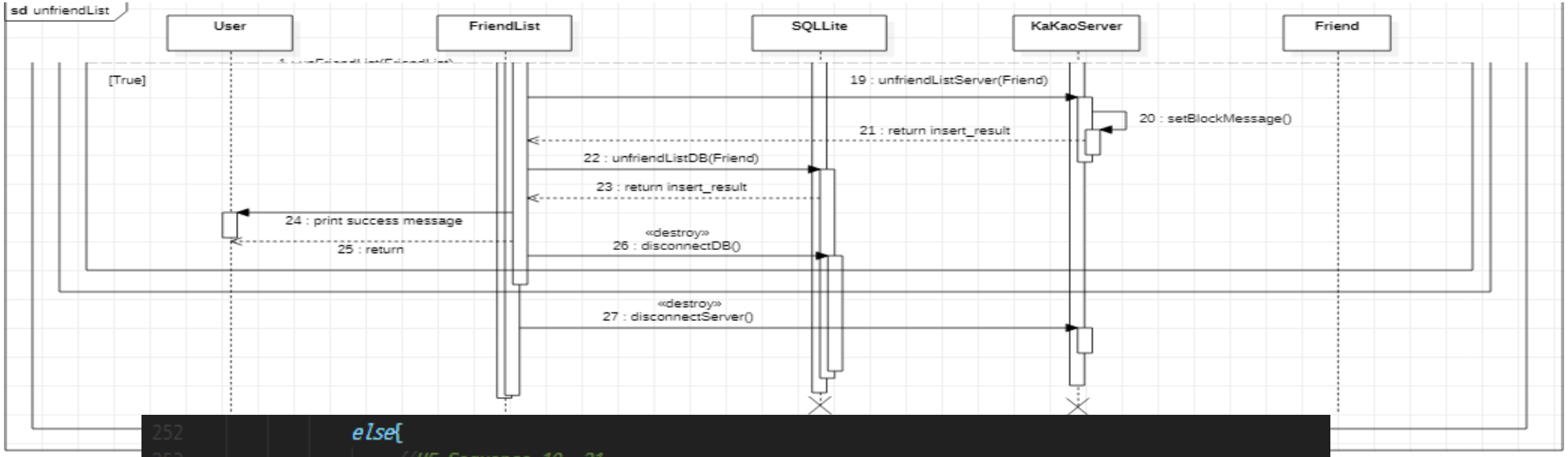
# 02. 기능별 구현 – U5.친구 차단(3 / 4)




FriendList.java

```
244             else{
245                 /* U5.15: U1, U2 와 다르게 서버와 데이터베이스를 Create 로 생성 , 통일성 X */
246                 boolean connectDB_result = sqlLite.connectDB();
247                 if(connectDB_result == false){
248                     System.out.println("Error!");    //U5.Sequence 17
249                     /* U5.18: 해당 부분을 실행하면 이후 26, 27 실행 불가, 따라서 삭제 필요 */
250                     /* return; */
251                 }
```

# 02. 기능별 구현 – U5.친구 차단(4 / 4)




JAVA

FriendList.java

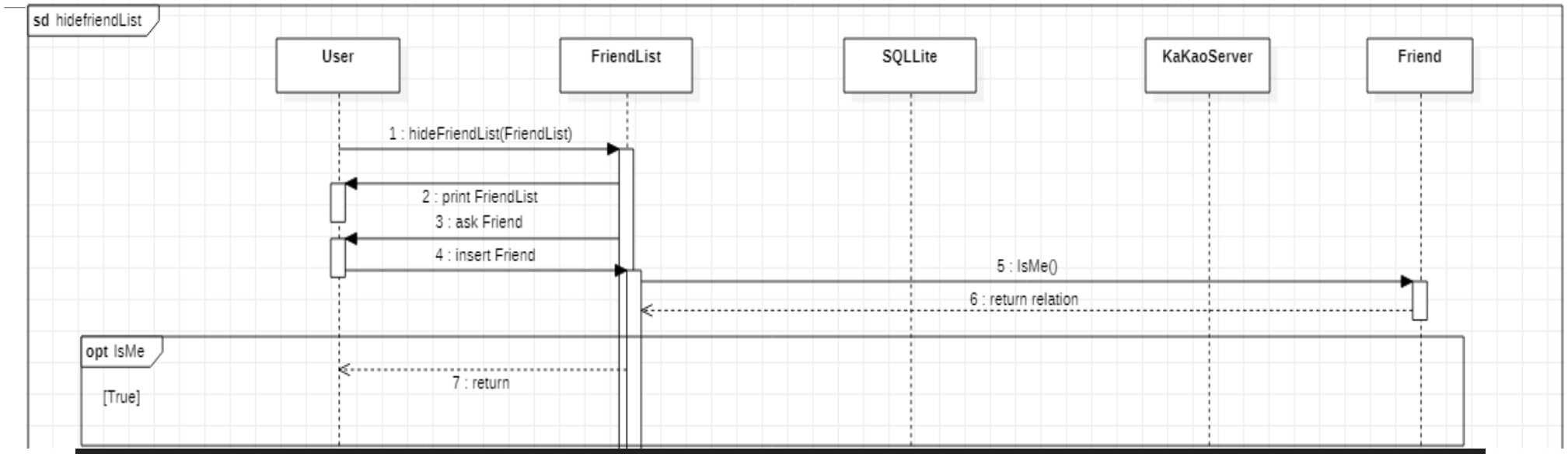
```
252                 else{
253                     //U5.Sequence 19, 21
254                     boolean insert_result1 = kaKaoServer.unFriendListServer(FriendList.get(friend));
255                     /* U5.21: setBlockMessage() 메소드의 리턴 값은 void 이지만, insert_result1 을 리턴함. */
256 
257                     //U5.Sequence 22, 23
258                     boolean insert_result2 = sqlLite.unFriendListDB(FriendList.get(friend));
259                     /* U5.24: 결과에 상관 없이 무조건 성공 메세지를 보냄. 아래와같이 if, else 로 수정함 */
260                     if(insert_result1 == true && insert_result2 == true){
261                         System.out.println("Success!");      //U5.Sequence24
262                         /* U5.25: 해당 부분을 실행하면 이후 26, 27 실행 불가, 따라서 삭제 필요 */
263                         /* return; */
264                     }
265                     sqlLite.disconnectDB();      //U5.Sequence 26
266                 }
267                 kaKaoServer.disconnectServer(); //U5.Sequence 27
268             }
269         }
270     }
```

# 02. 기능별 구현 – U6.친구 숨김(0 / 4)


FriendList.java

```java
public void hideFriendList(ArrayList<Friend> FriendList) {  //U6.Sequence 1
    for(int i = 0; i<FriendList.size(); i++){
        System.out.println(FriendList.get(i));              //U6.Sequence 2
    }
    System.out.println("Insert Friend");                    //U6.Sequence 3
    int friend = sc.nextInt();                              //U6.Sequence 4
    /* U6.4: Insert Friend 라는 메소드는 없는데 활성화를 가짐. */

    boolean relation = FriendList.get(friend).IsMe();       //U6.Sequence 5, 6
    if(relation == true){
        return;                                             //U6.Sequence 7
    }
    System.out.println("Insert hideFriendOption");          //U6.Sequence 8
    String hideFriendOption = sc.next();                    //U6.Sequence 9
    /* U6.9: Insert hideFriendOption 라는 메소드는 없는데 활성화를 가짐. */

    if(hideFriendOption.equals("Cancel")){
        return;
    }
    else if(hideFriendOption.equals("Agree")){
        /* U6.11: U1, U2 와 다르게 서버와 데이터베이스를 Create 로 생성 , 통일성 X */
        boolean connect_result = kaKaoServer.connectServer();
        if(connect_result == false) {
            System.out.println("Error!");                  //U6.Sequence 13
            return;
        }
        else{
            boolean connectDB_result = sqlLite.connectDB(); //U6.Sequence 15, 16
            if(connectDB_result == false){
                System.out.println("Error!");              //U6.Sequence 17
                return;
            }
            else {
                //U6.Sequence 19, 20
                boolean insert_result1 = kaKaoServer.hideFriendListServer(FriendList.get(friend));
                //U6.Sequence 21, 22
                boolean insert_result2 = sqlLite.hideFriendListDB(FriendList.get(friend));
                /* U6.23: 결과에 상관 없이 무조건 성공 메세지를 보냄. 아래와같이 if, else 로 수정함 */
                if(insert_result1 == true && insert_result2 == true){
                    System.out.println("Success!");       //U6.Sequence 23
                    /* U6.24: return 하게 되면, DB와 Server를 disconnect 못함. 따라서 return 부분 삭제*/
                    /* return; */
                }
                else {
                    System.out.println("Error!");
                }
                sqlLite.disconnectDB();         //U6.Sequence 25
            }
            kaKaoServer.disconnectServer();     //U6.Sequence 26
        }
    }
}
```

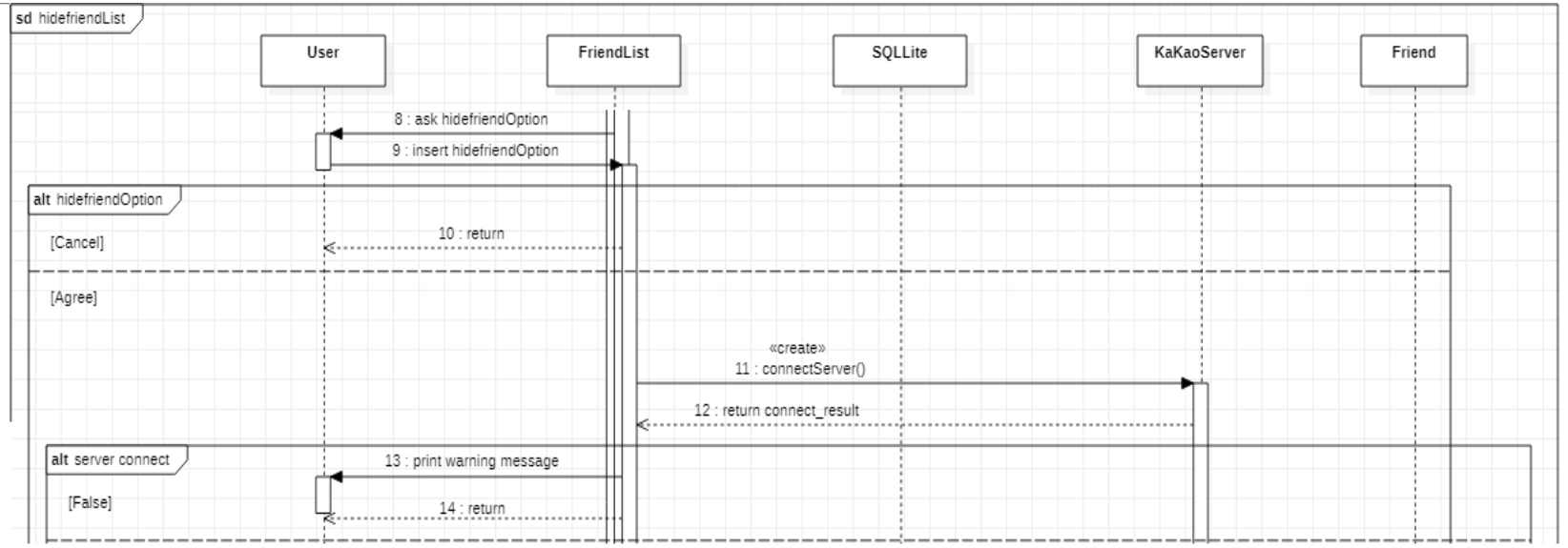
# 02. 기능별 구현 – U6.친구 숨김(1 / 4)




FriendList.java

```java
public void hideFriendList(ArrayList<Friend> FriendList) {  //U6.Sequence 1
    for(int i = 0; i<FriendList.size(); i++){
        System.out.println(FriendList.get(i));              //U6.Sequence 2
    }
    System.out.println("Insert Friend");                    //U6.Sequence 3
    int friend = sc.nextInt();                              //U6.Sequence 4
    /* U6.4: Insert Friend 라는 메소드는 없는데 활성화를 가짐. */

    boolean relation = FriendList.get(friend).IsMe();       //U6.Sequence 5, 6
    if(relation == true){
        return;                                             //U6.Sequence 7
    }
```

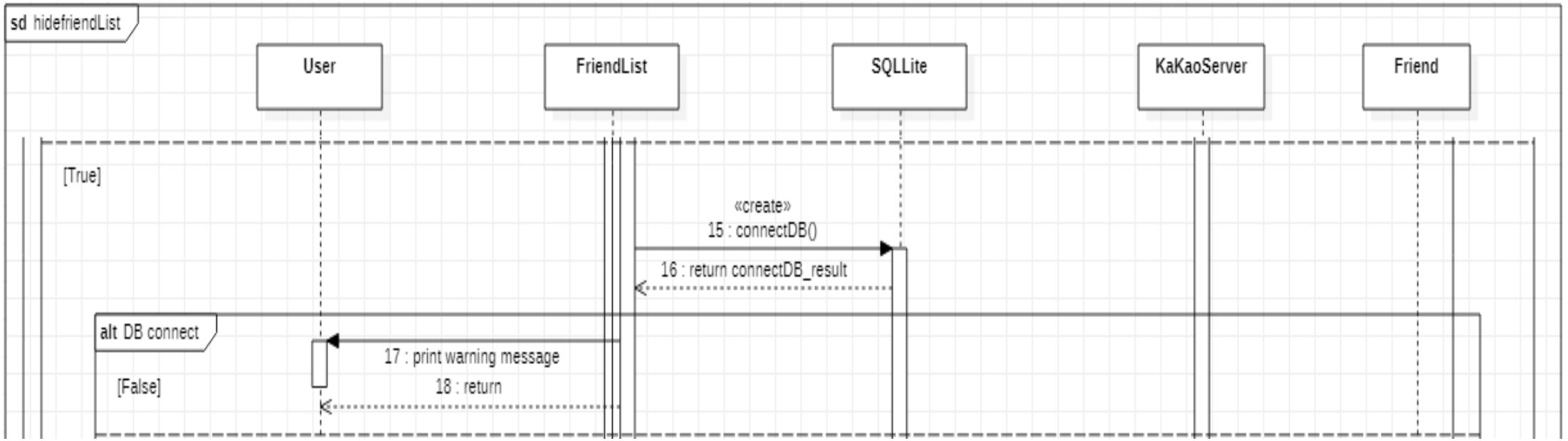
# 02. 기능별 구현 – U6.친구 숨김(2 / 4)




FriendList.java

```java
System.out.println("Insert hideFriendOption");          //U6.Sequence 8
String hideFriendOption = sc.next();                    //U6.Sequence 9
/* U6.9: Insert hideFriendOption 라는 메소드는 없는데 활성화를 가짐. */

if(hideFriendOption.equals("Cancel")){
    return;
}
else if(hideFriendOption.equals("Agree")){
    /* U6.11: U1, U2 와 다르게 서버와 데이터베이스를 Create 로 생성 , 통일성 X */
    boolean connect_result = kaKaoServer.connectServer();
    if(connect_result == false) {
        System.out.println("Error!");                  //U6.Sequence 13
        return;
    }
```

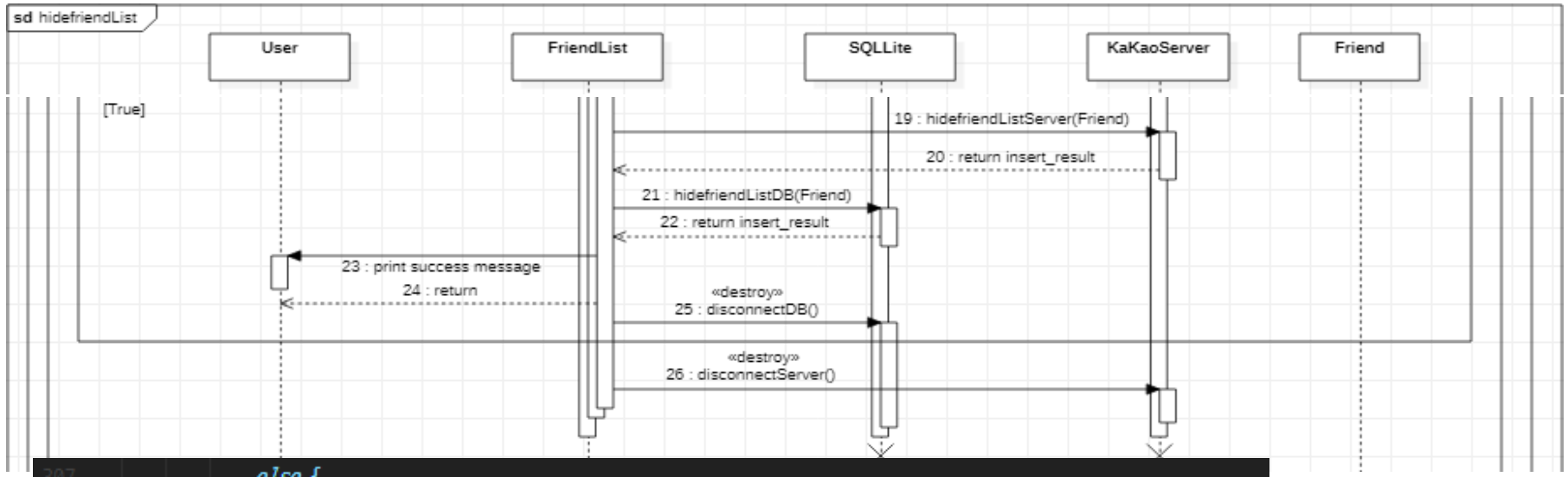
# 02. 기능별 구현 – U6.친구 숨김(3 / 4)




FriendList.java

```
301         else{
302             boolean connectDB_result = sqlLite.connectDB(); //U6.Sequence 15, 16
303             if(connectDB_result == false){
304                 System.out.println("Error!");              //U6.Sequence 17
305                 return;
306             }
```

# 02. 기능별 구현 – U6.친구 숨김(4 / 4)




JAVA

FriendList.java

```
307                 else {
308                     //U6.Sequence 19, 20
309                     boolean insert_result1 = kaKaoServer.hideFriendListServer(FriendList.get(friend));
310                     //U6.Sequence 21, 22
311                     boolean insert_result2 = sqlLite.hideFriendListDB(FriendList.get(friend));
312                     /* U6.23: 결과에 상관 없이 무조건 성공 메세지를 보냄. 아래와같이 if, else 로 수정함 */
313                     if(insert_result1 == true && insert_result2 == true){
314                         System.out.println("Success!");          //U6.Sequence 23
315                         /* U6.24: return 하게 되면, DB와 Server를 disconnect 못함. 따라서 return 부분 삭제*
316                         /* return; */
317                     }
318                     else {
319                         System.out.println("Error!");
320                     }
321                     sqlLite.disconnectDB();          //U6.Sequence 25
322                 }
323                 kaKaoServer.disconnectServer();      //U6.Sequence 26
324             }
325         }
326     }
```

# 02. 기능별 구현 – U7.친구 목록 복구 (0 / 6)

FriendList.java

option()

```java
//---------------------------------- U7 친구 목록 복구 ----------------------------------
public void option() {                          //U7.Sequence 1
    System.out.println("Insert Option");        //U7.Sequence 2
    String strOption = sc.next();               //U7.Sequence 3

    boolean connect_result = kaKaoServer.connectServer();   //U7.Sequence 4, 5
    if(connect_result == false){
        System.out.println("Error!");          //U7.Sequence 6
    }
    else{
        boolean connectDB_result = sqlLite.connectDB(); //U7.Sequence 7, 8
        if(connectDB_result == false){
            System.out.println("Error!");      //U7.Sequence 9
        }
        else{
            if(strOption.equals("숨김목록복구")){
                backHideFriend();               //U7.Sequence 10
            }
            else if(strOption.equals("차단목록복구")){
                backBlockFriend();              //U7.Sequence 25
            }
        }
    }
}
```

숨김 목록 복구

backHideFriend()

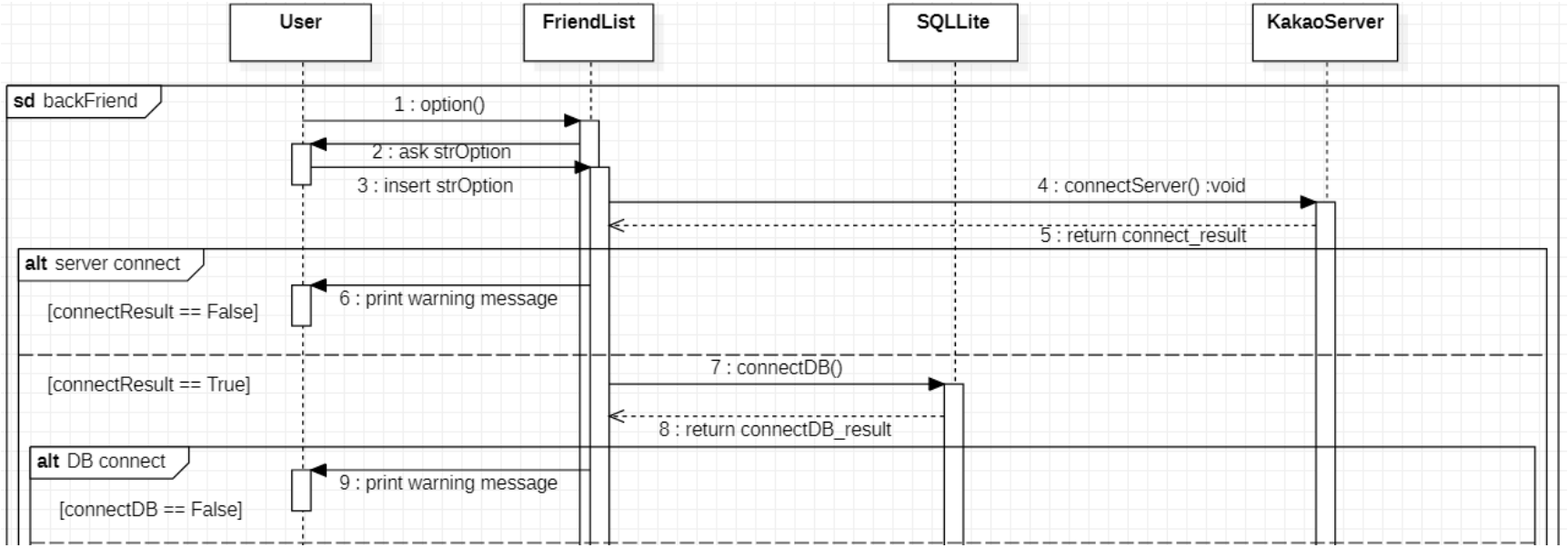
```java
public void backHideFriend() {
    hideFriendList = sqlLite.getHideFriendList();   //U7.Sequence 11, 12
    for(int i = 0; i < hideFriendList.size(); i++) {
        System.out.println(hideFriendList.get(i));  //U7.Sequence 13
    }
    int friend = sc.nextInt();                  //U7.Sequence 14

    System.out.println("SelectOption 1, 2, 3"); //U7.Sequence 15
    int option = sc.nextInt();                  //U7.Sequence 16
    /* 해당부분이후 다이어그램에 Sequence 번호가 잘못됨. */

    if(option == 1){
        sqlLite.setHFriendInform(hideFriendList.get(friend));   //U7.Sequence 17
    }
    else if(option == 2){
        /* U7.18: blockUserMessage 메소드는 클래스다이어그램에 없음. */
        //U7.Sequence 18, 19
        boolean result = kaKaoServer.blockUserMessage(hideFriendList.get(friend));
        sqlLite.setBFriendInform(hideFriendList.get(friend));   //U7.Sequence 20
    }
    else if(option == 3){
        sqlLite.deleteFriend(hideFriendList.get(friend));       //U7.Sequence 21
    }
    kaKaoServer.disconnectServer();         //U7.Sequence 22
    sqlLite.disconnectDB();                 //U7.Sequence 23

    System.out.println("Success!");         //U7.Sequence 24
}
```

차단 목록 복구

backBlockFriend()

```java
public void backBlockFriend() {
    unFriendList = sqlLite.getUnFriendList();       //U7.Sequence 26, 27
    for(int i = 0; i < unFriendList.size(); i++) {
        System.out.println(unFriendList.get(i));    //U7.Sequence 28
    }
    int friend = sc.nextInt();                      //U7.Sequence 29

    //U7.Sequence 30, 31
    /* U7.30: backBlockUser 메소드는 클래스다이어그램에 없음. */
    boolean result1 = kaKaoServer.backBlockUser(unFriendList.get(friend));
    //U7.Sequence 32, 33
    /* U7.32: setBFriendInform 메소드 클래스다이어그램에서는 void, 통일성 X */
    boolean result2 = sqlLite.setBFriendInform(unFriendList.get(friend));
    kaKaoServer.disconnectServer();         //U7.Sequence 34
    sqlLite.disconnectDB();                 //U7.Sequence 35
    /* U7.36: 설명은 result 값이 false 일때는 성공메세지를 보내지 않는다고 되어 있으나 */
    /* 다이어그램에는 해당 내용 없음. 통일성 X, 설명대로 구현하였음. */
    if(result1 == true && result2 == true){
        System.out.println("Success!");
    }
}
```

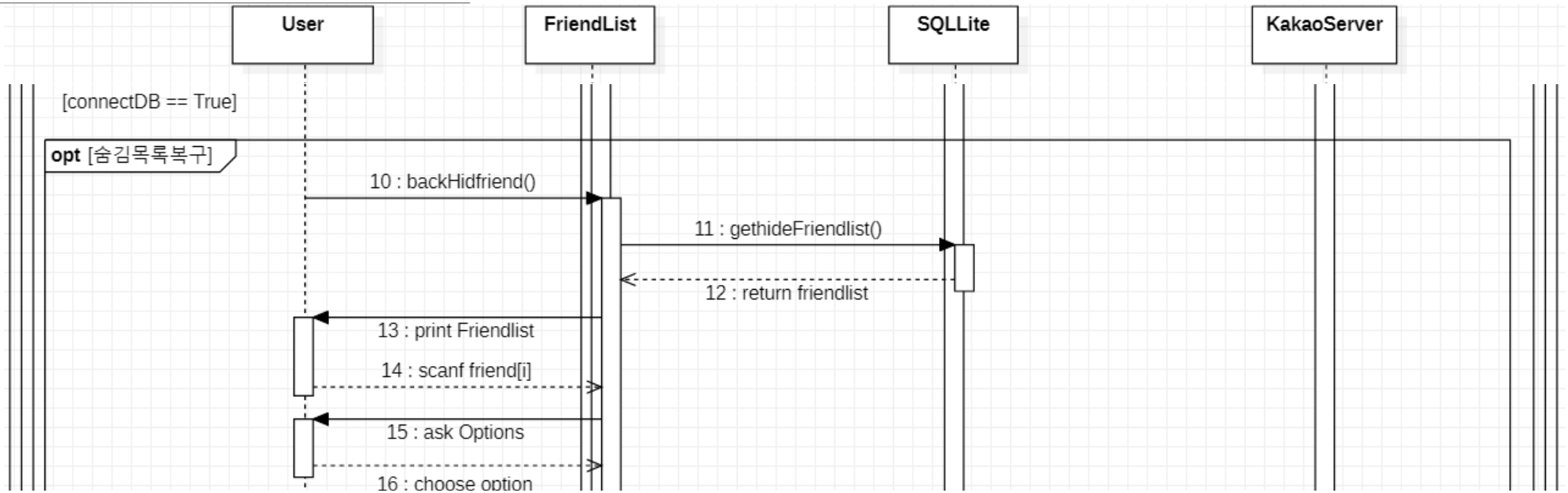
# 02. 기능별 구현 – U7.친구 목록 복구 (1 / 6)




FriendList.java

```java
//---------------------------------- U7 친구 목록 복구 ----------------------
public void option() {                          //U7.Sequence 1
    System.out.println("Insert Option");       //U7.Sequence 2
    String strOption = sc.next();               //U7.Sequence 3

    boolean connect_result = kaKaoServer.connectServer();    //U7.Sequence 4, 5
    if(connect_result == false){
        System.out.println("Error!");          //U7.Sequence 6
    }
    else{
        boolean connectDB_result = sqlLite.connectDB(); //U7.Sequence 7, 8
        if(connectDB_result == false){
            System.out.println("Error!");      //U7.Sequence 9
        }
```

# 02. 기능별 구현 – U7.친구 목록 복구 (2 / 6)

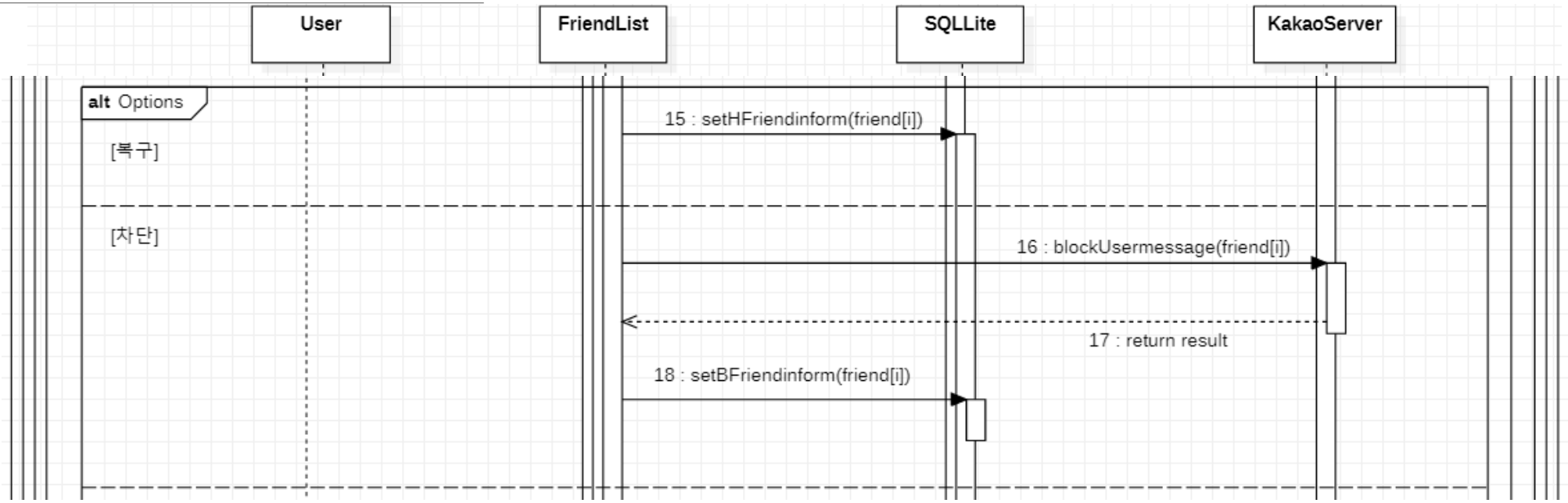



FriendList.java

```
343             else{
344                 if(strOption.equals("숨김목록복구")){
345                     backHideFriend();                    //U7.Sequence 10
346                 }
```

```
354     public void backHideFriend() {
355         hideFriendList = sqlLite.getHideFriendList();    //U7.Sequence 11, 12
356         for(int i = 0; i < hideFriendList.size(); i++) {
357             System.out.println(hideFriendList.get(i));   //U7.Sequence 13
358         }
359         int friend = sc.nextInt();                       //U7.Sequence 14
360 
361         System.out.println("SelectOption 1, 2, 3"); //U7.Sequence 15
362         int option = sc.nextInt();                       //U7.Sequence 16
363         /* 해당부분이후 다이어그램에 Sequence 번호가 잘못됨. */
364 
```

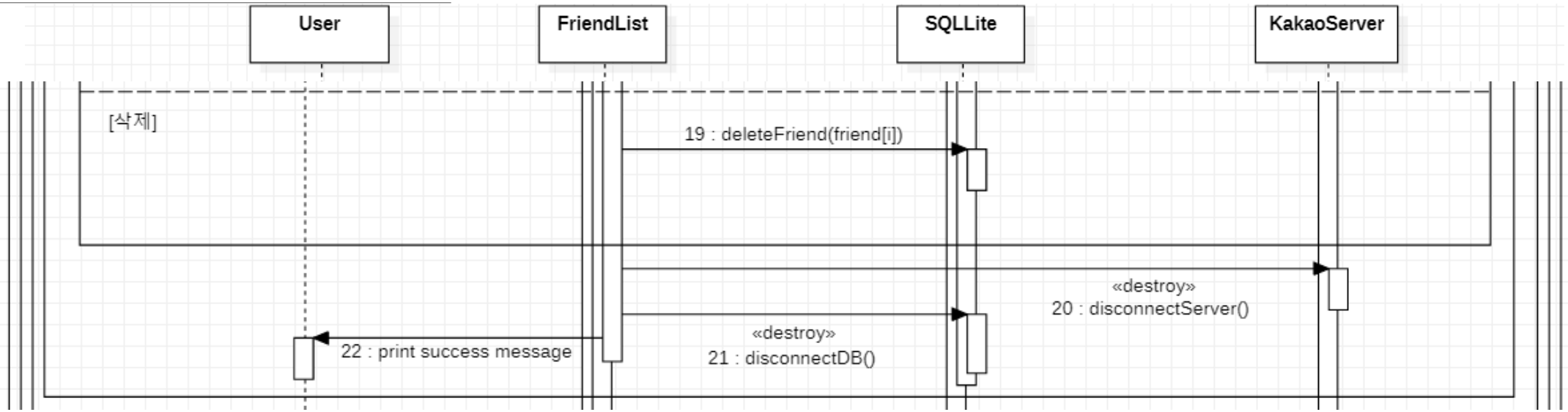
# 02. 기능별 구현 – U7.친구 목록 복구 (3 / 6)




FriendList.java

```java
if(option == 1){
    sqlLite.setHFriendInform(hideFriendList.get(friend));    //U7.Sequence 17
}
else if(option == 2){
    /* U7.18: blockUserMessage 메소드는 클래스다이어그램에 없음. */
    //U7.Sequence 18, 19
    boolean result = kaKaoServer.blockUserMessage(hideFriendList.get(friend));
    sqlLite.setBFriendInform(hideFriendList.get(friend));    //U7.Sequence 20
}
```

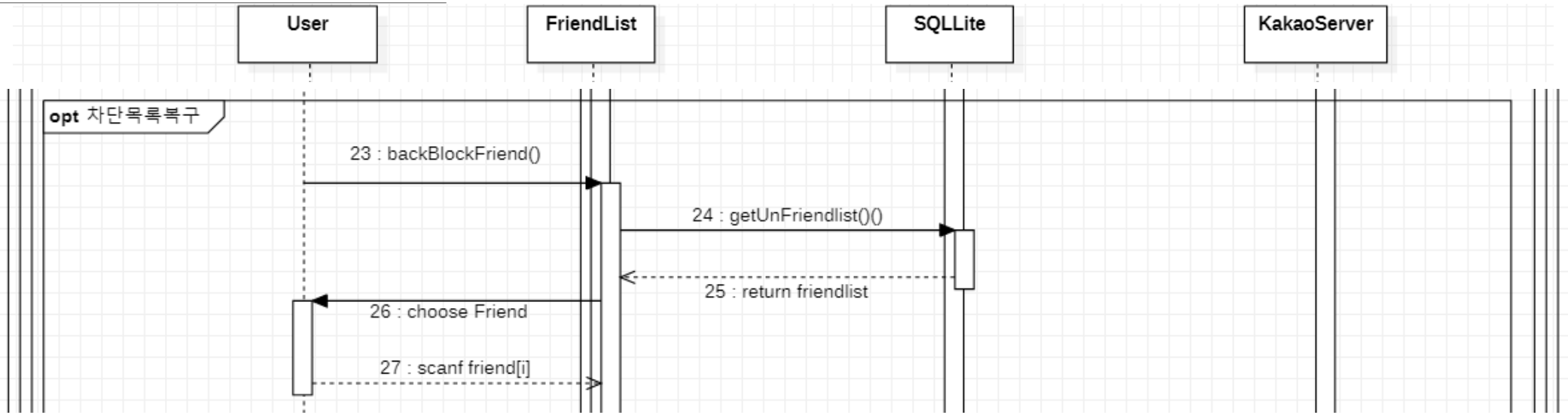
# 02. 기능별 구현 – U7.친구 목록 복구 (4 / 6)




FriendList.java

```
374         else if(option == 3){
375             sqlLite.deleteFriend(hideFriendList.get(friend));       //U7.Sequence 21
376         }
377         kaKaoServer.disconnectServer();         //U7.Sequence 22
378         sqlLite.disconnectDB();                 //U7.Sequence 23
379
380         System.out.println("Success!");         //U7.Sequence 24
381     }
```

# 02. 기능별 구현 – U7.친구 목록 복구 (5 / 6)

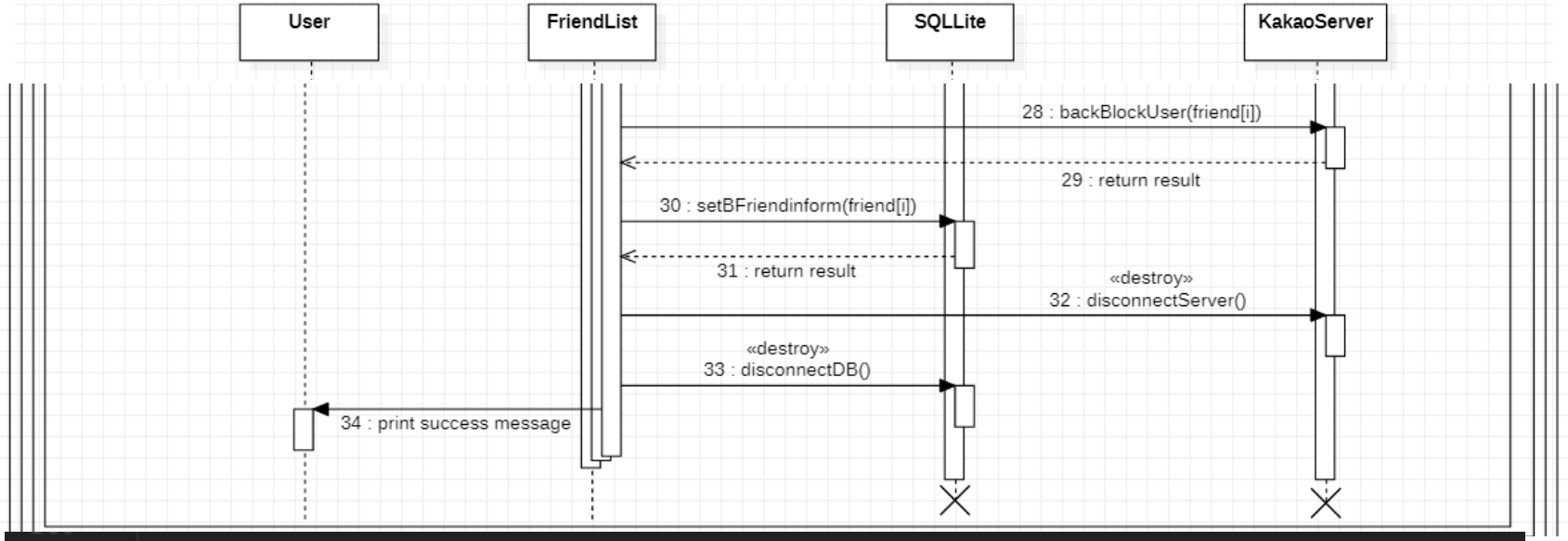



FriendList.java

```
343         else{
344             if(strOption.equals("숨김목록복구")){
345                 backHideFriend();              //U7.Sequence 10
346             }
347             else if(strOption.equals("차단목록복구")){
348                 backBlockFriend();             //U7.Sequence 25
349             }
350         }
```

```
383     public void backBlockFriend() {
384         unFriendList = sqlLite.getUnFriendList();        //U7.Sequence 26, 27
385         for(int i = 0; i < unFriendList.size(); i++) {
386             System.out.println(unFriendList.get(i));     //U7.Sequence 28
387         }
388         int friend = sc.nextInt();                       //U7.Sequence 29
389
```

# 02. 기능별 구현 – U7.친구 목록 복구 (6 / 6)




JAVA

FriendList.java

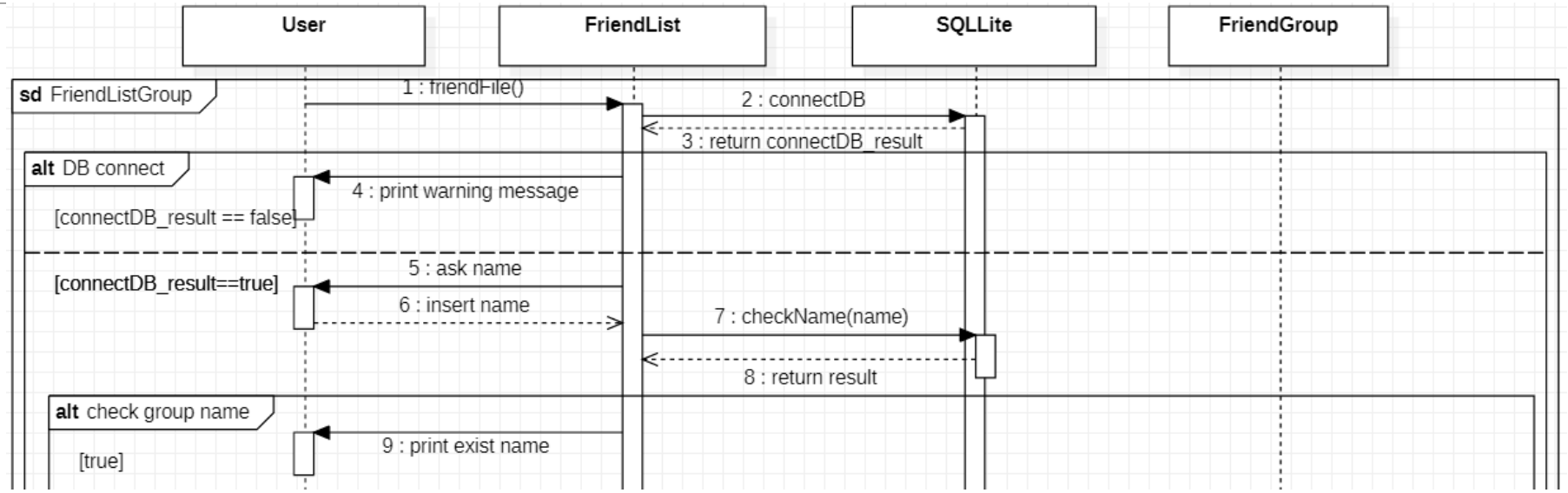
```
390         //U7.Sequence 30, 31
391         /* U7.30: backBlockUser 메소드는 클래스다이어그램에 없음. */
392         boolean result1 = kaKaoServer.backBlockUser(unFriendList.get(friend));
393         //U7.Sequence 32, 33
394         /* U7.32: setBFriendInform 메소드 클래스다이어그램에서는 void, 통일성 X */
395         boolean result2 = sqlLite.setBFriendInform(unFriendList.get(friend));
396         kaKaoServer.disconnectServer();          //U7.Sequence 34
397         sqlLite.disconnectDB();                  //U7.Sequence 35
398         /* U7.36: 설명은 result 값이 false 일때는 성공메세지를 보내지 않는다고 되어 있으나
399         /* 다이어그램에는 해당 내용 없음. 통일성 X, 설명대로 구현하였음. */
400         if(result1 == true  && result2 == true){
401             System.out.println("Success!");
402         }
403     }
```

# 02. 기능별 구현 – U8.친구 목록 그룹화 (0 / 4)


FriendList.java

```java
//---------------------------------- U8 친구 목록 그룹화 ----------------------------------------------------
public void friendFile() {                                  //U8.Sequence 1
    boolean connectDB_result = sqlLite.connectDB(); //U8.Sequence 2, 3
    if (connectDB_result == false){
        System.out.println("Warning!");      //U8.Sequence 4
    }
    else{
        System.out.println("Insert name");   //U8.Sequence 5
        String name = sc.next();             //U8.Sequence 6

        /* U8.7: checkName 메소드는 클래스다이어그램에 없음. */
        boolean result = sqlLite.checkName(name);    //U8.Sequence 7, 8
        if(result == true){
            System.out.println("Exist name!");       //U8.Sequence 9
        }
        else{
            FriendGroup friendGroup = new FriendGroup(name); //U8.Sequence 10
            friendList = sqlLite.getFriendList();            //U8.Sequence 11
            for(int i = 0; i < friendList.size(); i++){
                System.out.println(friendList.get(i));       //U8.Sequence 12
            }
            System.out.println("Insert number");     //U8.Sequence 13
            int number = sc.nextInt();               //U8.Sequence 14
            for(int i = 0; i<number; i++){
                String friend = sc.next();           //U8.Sequence 15
                if(sqlLite.checkName(friend) == false){
                    for(int j = 0; j < friendList.size(); j++){
                        if(friendList.get(j).getStrName() == friend){
                            friendGroup.addGroupFriend(friendList.get(j));  //U8.Sequence 16
                        }
                    }
                }
                else{
                    System.out.println("Exist Friends"); //U8.Sequence 17
                }
            }
            /* U8.18: 친구 그룹을 추가 하는데, 이름만 등록함. friendGroup도 같이 저장해야 한다고 생각함.*/
            /* 잘못된 부분: sqlLite.saveFriendGroup(name); */
            sqlLite.saveFriendGroup(friendGroup, name);  //U8.Sequence 18
            friendGroup = null;                          //U8.Sequence 19
            sqlLite.disconnectDB();                      //U8.Sequence 20
        }
    }
}
```

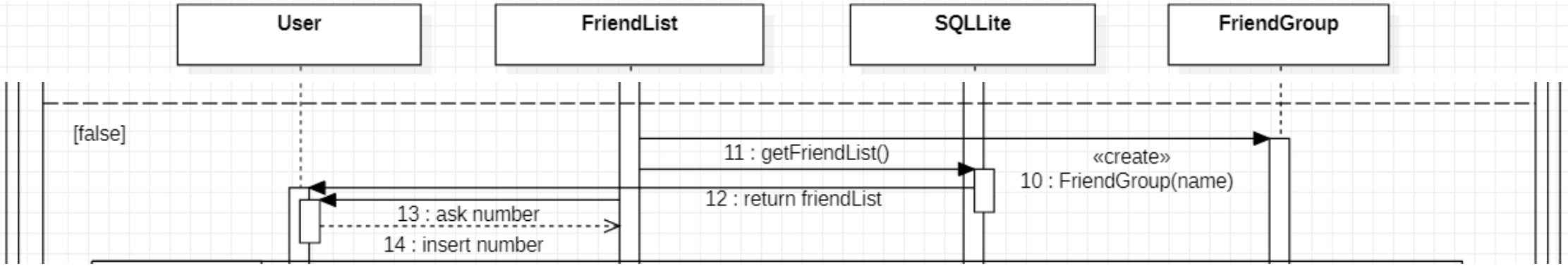
# 02. 기능별 구현 – U8.친구 목록 그룹화 (1 / 4)




FriendList.java

```java
406    //------------------------------------ U8 친구 목록 그룹화 ----------------------------------------
407    public void friendFile() {                                  //U8.Sequence 1
408        boolean connectDB_result = sqlLite.connectDB(); //U8.Sequence 2, 3
409        if (connectDB_result == false){
410            System.out.println("Warning!");      //U8.Sequence 4
411        }
412        else{
413            System.out.println("Insert name");   //U8.Sequence 5
414            String name = sc.next();             //U8.Sequence 6
415
416            /* U8.7: checkName 메소드는 클래스다이어그램에 없음. */
417            boolean result = sqlLite.checkName(name);    //U8.Sequence 7, 8
418            if(result == true){
419                System.out.println("Exist name!");       //U8.Sequence 9
420            }
```

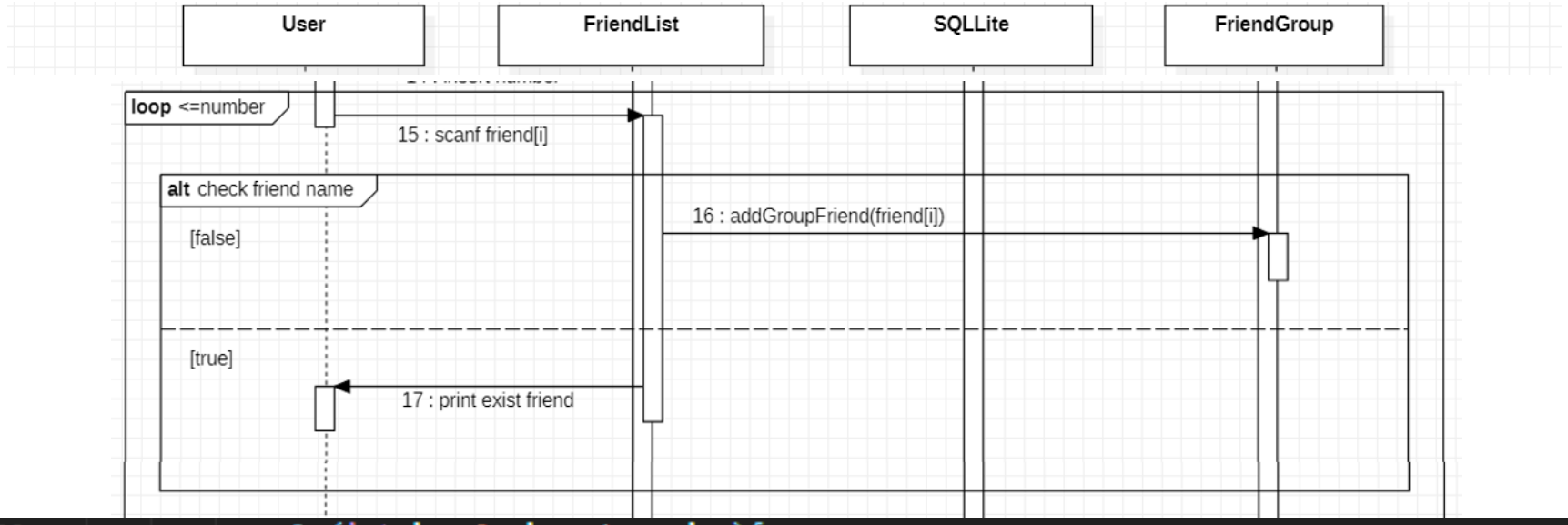
# 02. 기능별 구현 – U8.친구 목록 그룹화 (2 / 4)




FriendList.java

```
421         else{
422             FriendGroup friendGroup = new FriendGroup(name);//U8.Sequence 10
423             friendList = sqlLite.getFriendList();           //U8.Sequence 11
424             for(int i = 0; i < friendList.size(); i++){
425                 System.out.println(friendList.get(i));      //U8.Sequence 12
426             }
427             System.out.println("Insert number");     //U8.Sequence 13
428             int number = sc.nextInt();               //U8.Sequence 14
```

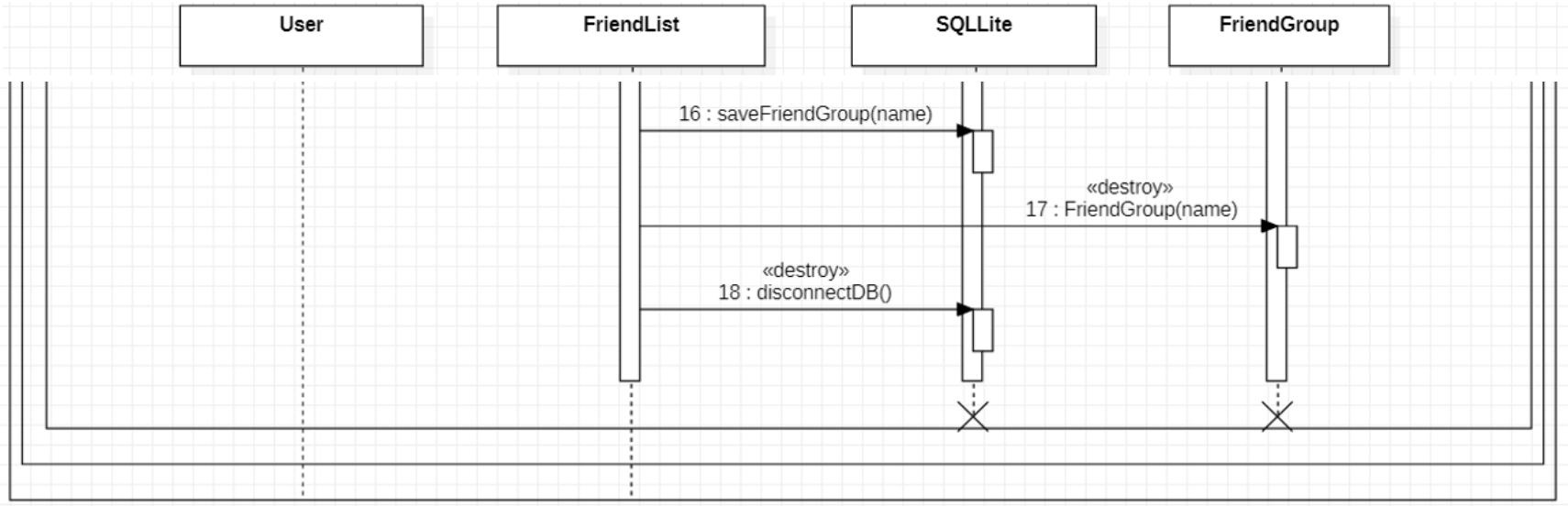
# 02. 기능별 구현 – U8.친구 목록 그룹화 (3 / 4)




FriendList.java

```
429             for(int i = 0; i<number; i++){
430                 String friend = sc.next();          //U8.Sequence 15
431                 if(sqlLite.checkName(friend) == false){
432                     for(int j = 0; j < friendList.size(); j++){
433                         if(friendList.get(j).getStrName() == friend){
434                             friendGroup.addGroupFriend(friendList.get(j));  //U8.Sequence 16
435                         }
436                     }
437                 }
438                 else{
439                     System.out.println("Exist Friends"); //U8.Sequence 17
440                 }
441             }
```

# 02. 기능별 구현 – U8.친구 목록 그룹화 (4 / 4)




FriendList.java

```
442                 /* U8.18: 친구 그룹을 추가 하는데, 이름만 등록함. friendGroup도 같이 저장해야 한다고 생각함.*/
443                 /* 잘못된 부분: sqlLite.saveFriendGroup(name); */
444                 sqlLite.saveFriendGroup(friendGroup, name);   //U8.Sequence 18
445                 friendGroup = null;                           //U8.Sequence 19
446                 sqlLite.disconnectDB();                       //U8.Sequence 20
447             }
448         }
449     }
```

# 02. 기능별 구현 – 그외 구현을 위한 필요한 코드 (1 / 5)


Friend.java

```java
public class Friend {
    private String strName;
    /* 클래스다이어그램에는 String 이지만, 추후 사용처는 int */
    private int nPhoneNumber;
    public boolean relation;
    public Profile Profile;

    //--------------getter, setter --------------
    public String getStrName() {
        return strName;
    }
    public void setStrName(String strName) {
        this.strName = strName;
    }

    public int getnPhoneNumber() {
        return nPhoneNumber;
    }
    public void setnPhoneNumber(String strPhoneNumber) {
        this.nPhoneNumber = nPhoneNumber;
    }

    public boolean getRelation() {
        return relation;
    }
    public void setRelation(boolean relation) {
        this.relation = relation;
    }
```

| Friend |
| --- |
| -strName: String<br>-strPhoneNumber: String<br>+relation: boolean<br>+Profile: Profile |
| +Friend(strName, strPhoneNumber): void<br>+IsMe(): boolean |

```java
    public Profile getProfile() {
        return Profile;
    }
    public void setProfile(Profile profile) {
        Profile = profile;
    }
    //-----------------------------------------------

    public Friend(String strName, int nPhoneNumber){
        this.strName = strName;
        this.nPhoneNumber = nPhoneNumber;
    }

    public boolean IsMe(){
        return relation;
    }
}
```

# 02. 기능별 구현 – 그외 구현을 위한 필요한 코드 (2 / 5)


FriendGroup.java

```java
import java.util.ArrayList;

public class FriendGroup {
    private String strGroupName;
    public ArrayList<Friend> groupFriend;

    public FriendGroup(String name) {

    } /* 해당 부분 클래스다이어그램에는 입력하는 인자값 없음 */
    /* 시퀀스다이어그램에서는 String 값 입력, 통일성 X */

    public void addGroupFriend(Friend friend) {
        groupFriend.add(friend);
    }

    /*시퀀스에서 사용 X 통일성 X */
    public void deleteGroupFriend(Friend friend) {
        groupFriend.remove(friend);
    }
}
```

| FriendGroup |
| --- |
| -strGroupName: String<br>+groupFriend: ArrayList<Friend> |
| +FriendGroup()<br>+addGroupFriend(Friend): void<br>+deleteGroupFriend(Friend): void |

# 02. 기능별 구현 – 그외 구현을 위한 필요한 코드 (3 / 5)


KaKaoServer.java

```java
import java.util.ArrayList;

public class KaKaoServer {
    public boolean connectServer(){
        return true;
    }
    public boolean disconnectServer() {
        return true;
    }
    public Friend getFriendServer(int nPhoneNumber){
        Friend returnFriend = new Friend("name", 1231234);
        return returnFriend;
    }
    public boolean updateProfileServer(Friend friend, Profile profile) {
        return true;
    }
    public boolean hideFriendListServer(Friend friend) {
        return true;
    }
    public boolean unFriendListServer(Friend friend) {
        setBlockMessage();  //U5.Sequence 20
        return true;
    }
    public ArrayList<Friend> getFindFriendListServer(String strSearchWord) {
        ArrayList<Friend> returnArrayList = new ArrayList<>();
        return returnArrayList;
    }
```

| KakaoServer |
| --- |
| +connectServer(): boolean<br>+disconnectServer(): boolean<br>+getFriendServer(nPhoneNumber): ArrayList<Friend><br>+updateProfileServer(Friend, Profile): boolean<br>+hidefriendListServer(Friend): boolean<br>+unfriendListServer(Friend): boolean<br>+getFindFriendListServer(strSearchWord): ArrayList<Friend><br>+setblockMessage(): void |

```java
    public void setBlockMessage() {

    }
    public boolean blockUserMessage(Friend friend) {
        return true;
    }
    public boolean backBlockUser(Friend friend) {
        return true;
    }
}
```

# 02. 기능별 구현 – 그외 구현을 위한 필요한 코드 (4 / 5)

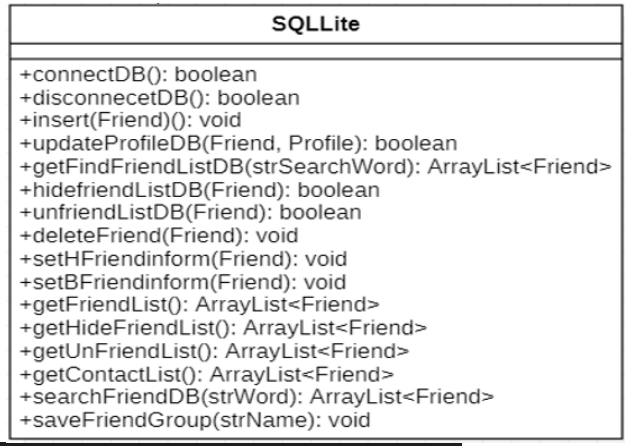

Profile.java

```java
import java.awt.*;

public class Profile {
    public String strName;
    public String strStatusMessage;
    public int nPImg;
    public Image nBImg[];

    public void showProfile() {

    }

    public void editProfile(String strName, String strStatusMessage, int nPImg, int nBImg) {
        /* 클래스다이어그램에는 입력 파라미터가 없는데, 시퀀스 다이어그램에서 4개의 변수 입력함. 통일성 X */
    }
}
```

| Profile |
| --- |
| +strName: String<br>+strStatusMessage: String<br>+nPImg: Int<br>+nBImg[]: Image |
| +showProfile(): void<br>+editProfile(): void |

# 02. 기능별 구현 – 그외 구현을 위한 필요한 코드 (5 / 5)


SQLLite.java




```java
import java.lang.reflect.Array;
import java.util.ArrayList;

public class SQLLite {
    public boolean connectDB() {
        return true;
    }
    public boolean disconnectDB() {
        return true;
    }

    /* 클래스다이어그램은 void, 하지만 추후 사용은 boolean, 통일성 X */
    public boolean Insert(Friend friend) {
        return true;
    }
    public boolean updateProfileDB(Friend friend, Profile profile) {
        return true;
    }
    public ArrayList<Friend> getFindFriendListDB(String strSearchWord){
        return null;
    }
    public boolean hideFriendListDB(Friend friend) {
        return true;
    }
    public boolean unFriendListDB(Friend friend) {
        return true;
    }
    public void deleteFriend(Friend friend) {
    }
    public void setHFriendInform(Friend friend) {
    }
```

```java
    /* 해당 부분 클래스다이어그램에는 void, 시퀀스다이어그램은 boolean, 통일성 X */
    public boolean setBFriendInform(Friend friend) {
        return true;
    }
    public ArrayList<Friend> getFriendList() {
        ArrayList<Friend> returnList = null;
        return returnList;
    }
    public ArrayList<Friend> getHideFriendList() {
        ArrayList<Friend> returnList = null;
        return returnList;
    }
    public ArrayList<Friend> getUnFriendList() {
        ArrayList<Friend> returnList = null;
        return returnList;
    }
    public ArrayList<Friend> getContactList() {
        ArrayList<Friend> returnList = null;
        return returnList;
    }
    /* //시퀀스에서 사용 X 통일성 */
    public ArrayList<Friend> searchFriendDB(String strWord) {
        ArrayList<Friend> returnList = null;
        return returnList;
    }
    /* 구현을 위해 FriendGroup 인자 추가 */
    public void saveFriendGroup(FriendGroup friendGroup, String strName) {
    }
    public boolean checkName(String name) {
        return true;
    }
}
```

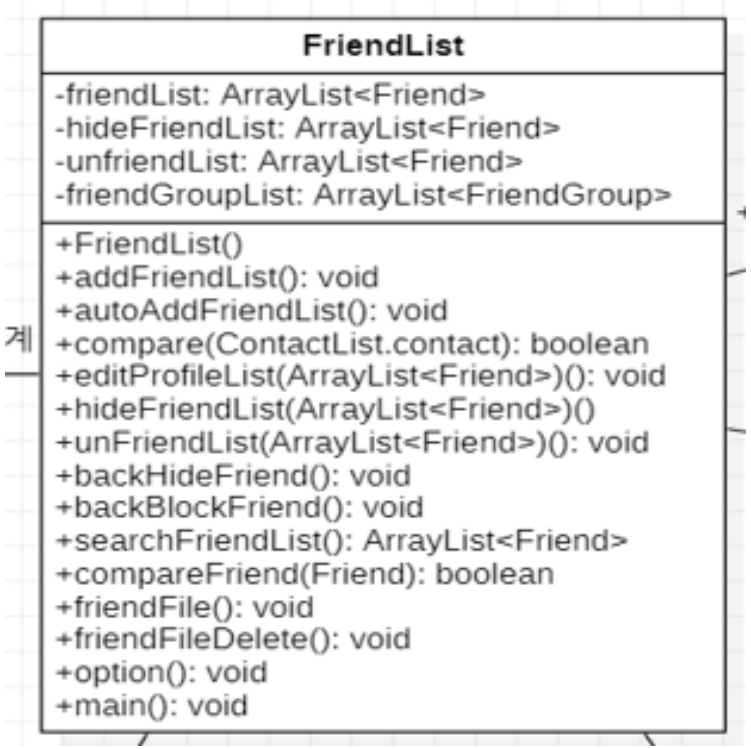
# 03

## 변경된 코드

# 03. 변경된 코드 – FriendList 클래스

FriendList.java

FriendList 클래스 다이어그램

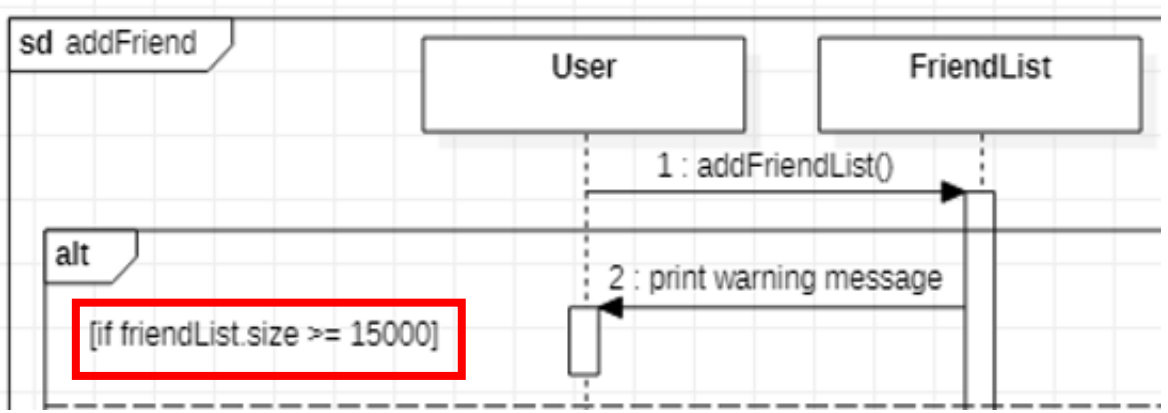



구현하기 위해 아래와 같은 맴버 변수가 필요.
kakaoServer, sqlLite

```
6   public class FriendList {
7       private ArrayList<Friend> friendList;
8       private ArrayList<Friend> hideFriendList;
9       private ArrayList<Friend> unFriendList;
10      private ArrayList<FriendGroup> friendGroupList;
11
12      KaKaoServer kaKaoServer;    /* 클래스 다이어그램에서 누락된 부분 */
13      SQLLite sqlLite;            /* 클래스 다이어그램에서 누락된 부분 */
```

# 03. 변경된 코드 – U1.친구추가 (1 / 2)

FriendList.java

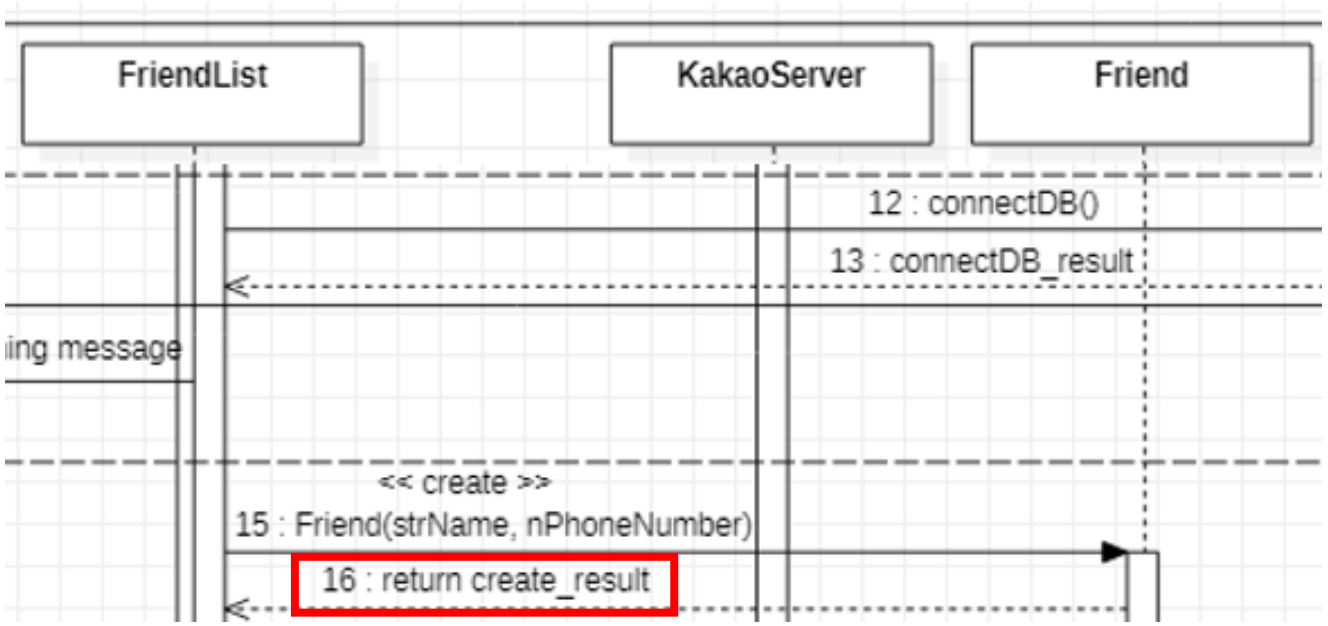



2. 만약 저장되어 있는 친구의 수가 15000명을 초과했다면, **경고 메시지를 출력**한다.

```
19      //------------------------------------ U1 친구 추가 --
20      public void addFriendList() {
21          if(friendList.size() > 15000) {
22              System.out.println("Error!");
23          }
```

# 03. 변경된 코드 – U1.친구추가 (2 / 2)

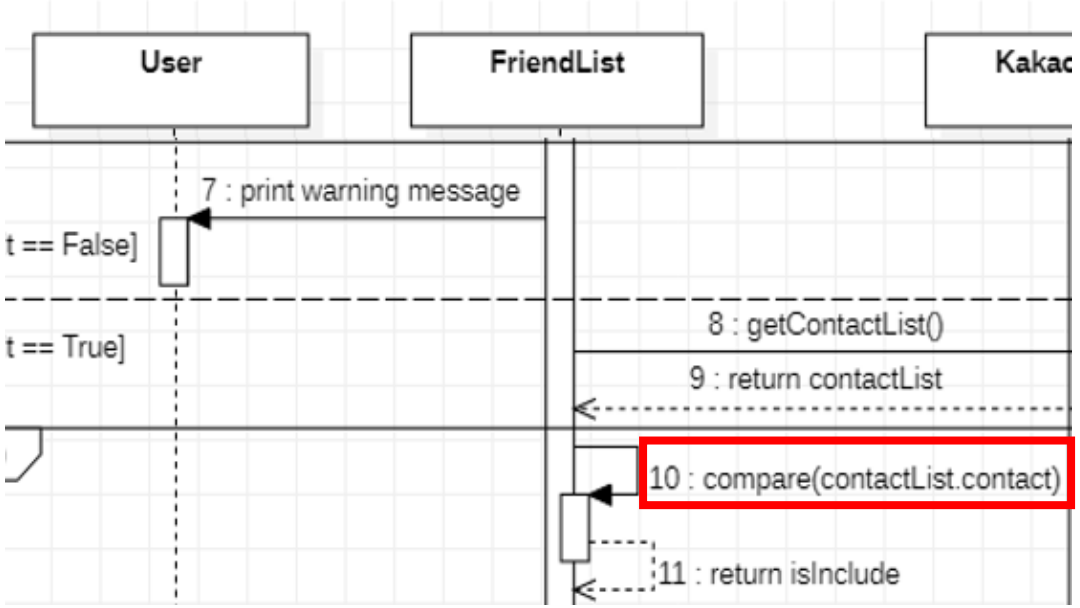
FriendList.java




```
if(connectDB_result == false){
    System.out.println("Error!");          //U1.Sequence 14
}
else{
    Friend friend = new Friend(strName, nPhoneNumber);   //U1.Sequence 15
    /* U1.Sequence 16, 생성자에는 return 이 없다. */
    boolean insert_result = sqlLite.Insert(friend);      //U1.Sequence 17, 18
```

# 03. 변경된 코드 – U2.자동친구추가 (1 / 2)

JAVA
FriendList.java




JAVA
Friend.java

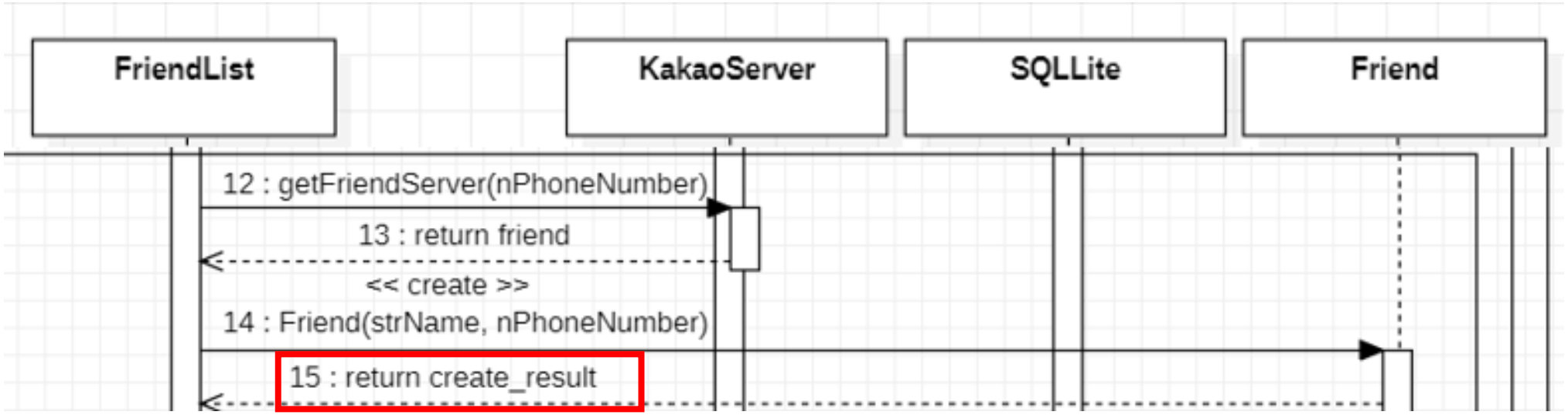
```java
public class Friend {
    private String strName;
    /* 클래스다이어그램에는 String 이지만, 추후 사용처는 int */
    private int nPhoneNumber;
    public boolean relation;
    public Profile Profile;
```

```java
for(int i = 0; i<contactList.size(); i ++){
    boolean isInclude = compare(contactList.get(i));   //U2.Sequence 10, 11
    if(isInclude == false){
```

```java
/* 시퀀스 다이어그램: ContactList.contact, 하지만 Friend 클래스에는 contact 맴버 변수 없음 */
public boolean compare(ArrayList<Friend> contactList) {
    boolean isInclude = true;        //U2.Sequence 10
    return true;                     //U2.Sequence 11
}
```

# 03. 변경된 코드 – U2.자동친구추가 (2 / 2)

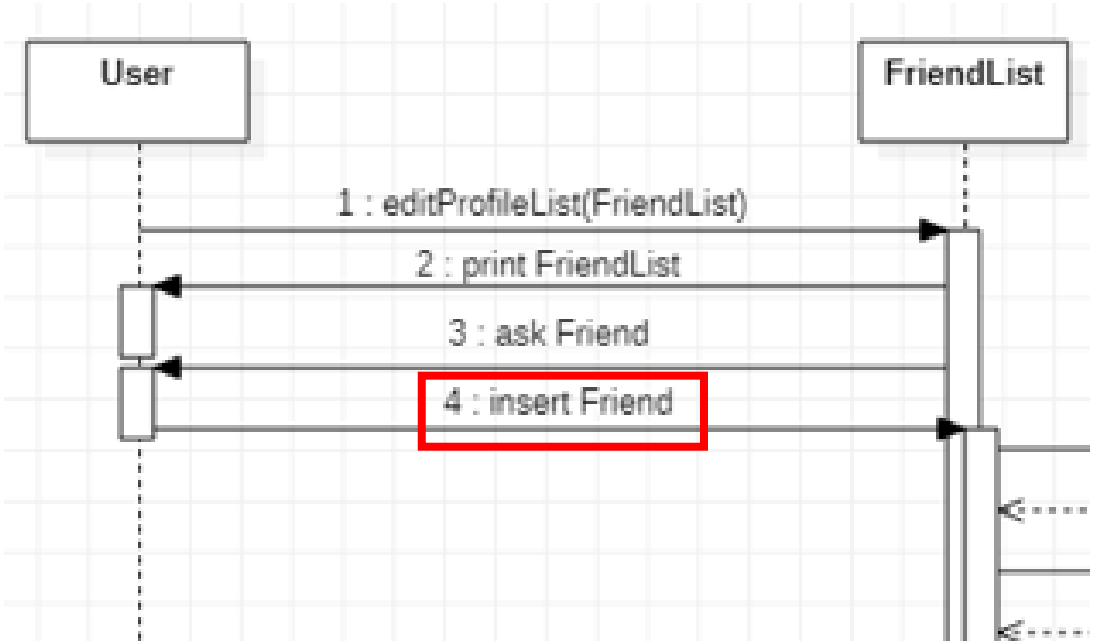
FriendList.java




```
if(isInclude == false){
    Friend friend = kaKaoServer.getFriendServer(contactList.get(i).getnPhoneNumber());
    //U2.Sequence 12, 13, 14
    /* U2.Sequence 15, 생성자는 리턴이 없음. */
    boolean insert_result = sqlLite.Insert(friend);      //U2.Sequence 16, 17
```

# 03. 변경된 코드 – U3.프로필 편집 (1 / 4)

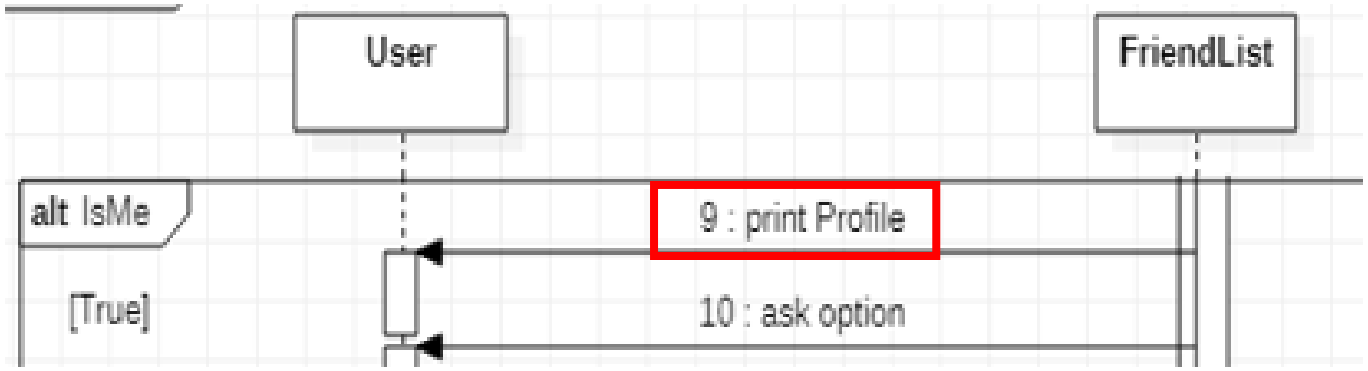
FriendList.java




```
System.out.println("Select Friend");                    //U3.Sequence 3
int Insert_Friend = sc.nextInt();                        //U3.Sequence 4
/* U3.4: Insert Friend 라는 메소드가 없는데, 활성화를 가짐 정정 필요 */
boolean relation = friendList.get(Insert_Friend).IsMe();          //U3.Sequence 5, 6
```

# 03. 변경된 코드 – U3.프로필 편집 (2 / 4)

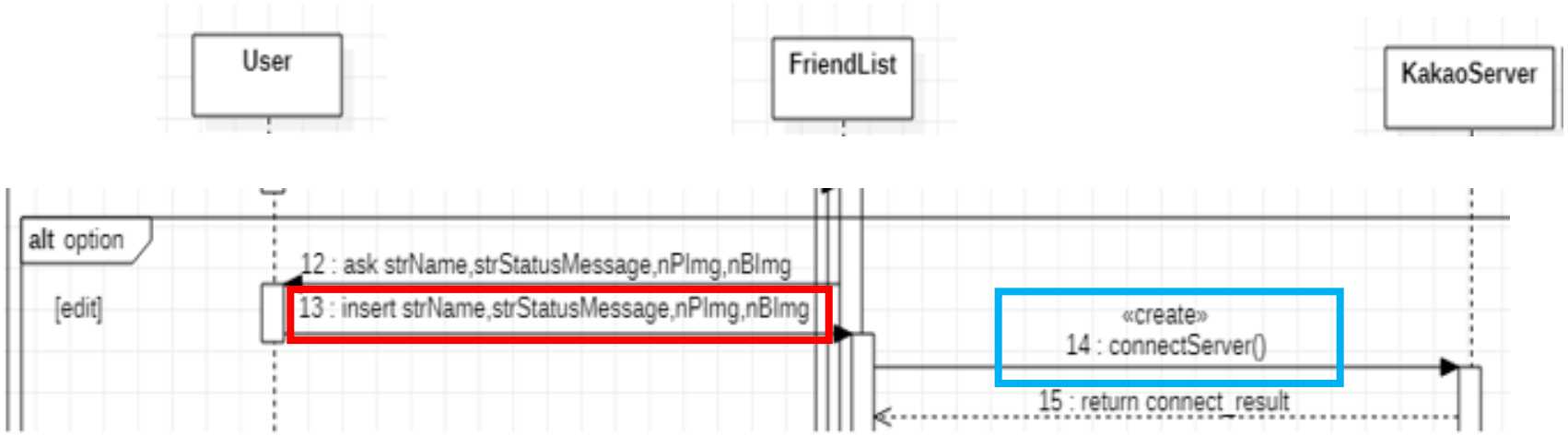
FriendList.java




```java
boolean relation = friendList.get(Insert_Friend).IsMe();                    //U3.Sequence 5, 6
Profile profile = friendList.get(Insert_Friend).getProfile();               //U3.Sequence 7, 8
if(relation == true){
    /* U3.9: print Profile 이라고 되어 있음. 메소드로 표현하는 것이 맞음. */
    profile.showProfile();                                                  //U3.Sequence 9
```

# 03. 변경된 코드 – U3.프로필 편집 (3 / 4)

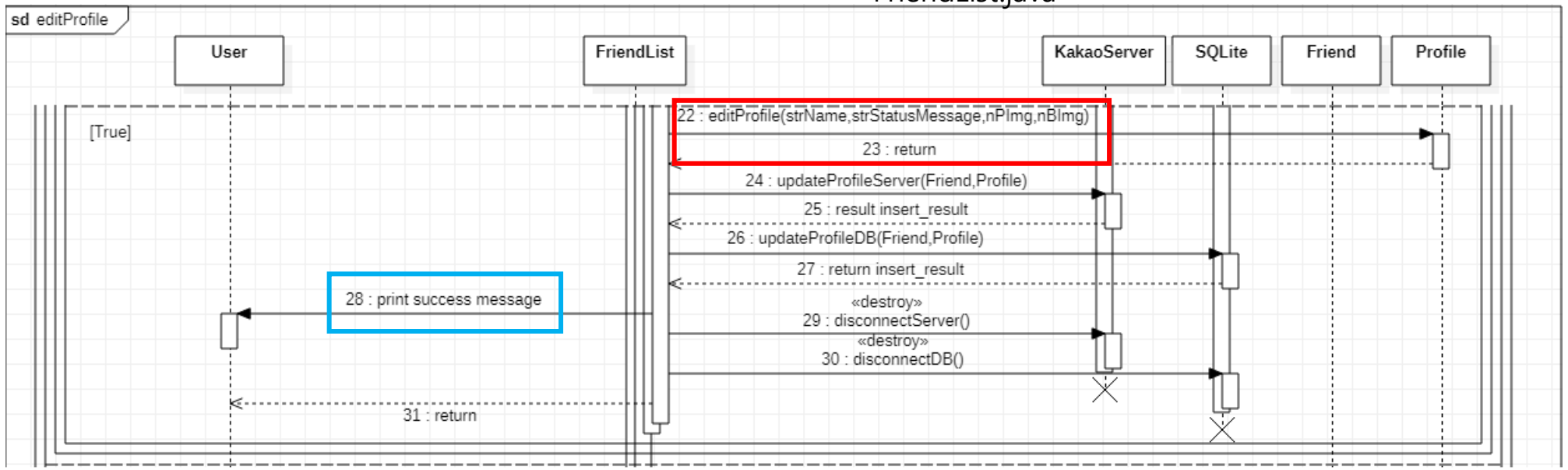
FriendList.java




```
System.out.println("Insert nBImg");                    //U3.Sequence 12
int nBImg = sc.nextInt();                               //U3.Sequence 13
/* U3.13: Insert Friend 라는 메소드가 없는데, 활성화를 가짐 정정 필요 */

/* U3.14: U1, U2 와 다르게 서버와 데이터베이스를 Create 로 생성 , 통일성 X */
boolean connect_result = kaKaoServer.connectServer();   //U3.Sequence 14, 15
if(connect_result == false){
    System.out.println("Error!");                      //U3.Sequence 16
}
```

# 03. 변경된 코드 – U3.프로필 편집 (4 / 4)

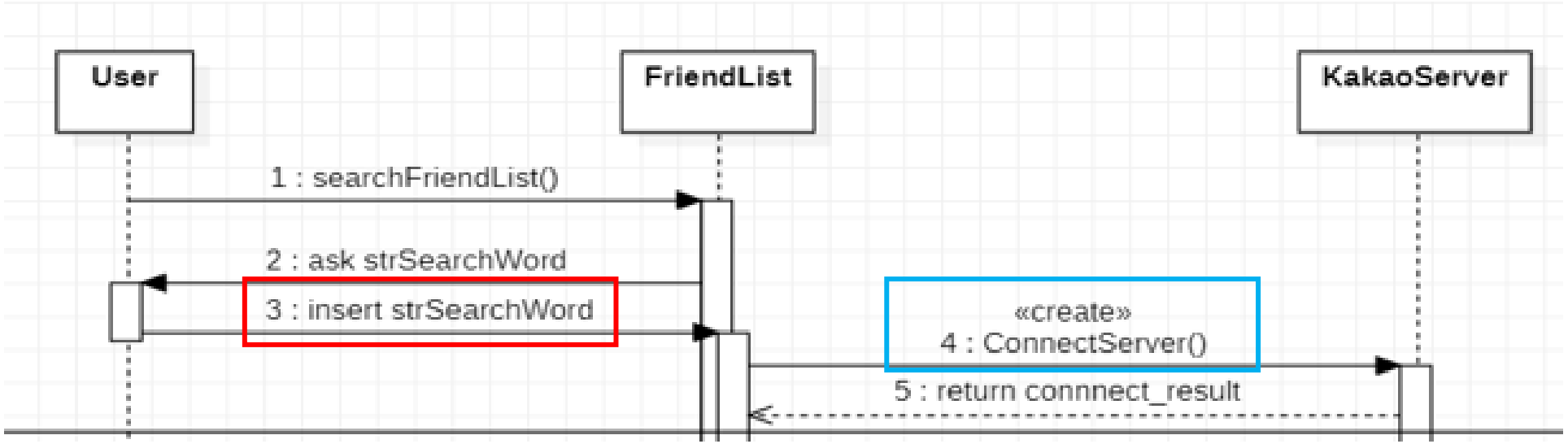
FriendList.java




```java
profile.editProfile(strName, strStatusMessage, nPImg, nBImg);    //U3.Sequence 22
/* U3.23: 클래스 다이어그램에 editProfile 은 void 형식, 리턴 불가능 */
boolean insert_result1 = kaKaoServer.updateProfileServer(friendList.get(Insert_Friend), profile);    //U3.Sequence 24, 25
boolean insert_result2 = sqlLite.updateProfileDB(friendList.get(Insert_Friend), profile);        //U3.Sequence 26, 27
/* U3.28: 결과에 상관 없이 무조건 성공 메세지를 보냄. 아래와같이 if, else 로 수정함 */
if(insert_result1 == true && insert_result2 == true) {   //U3.Sequence 28
    System.out.println("Success!");
}
else{
    System.out.println("Error!");
}
```

# 03. 변경된 코드 – U4.친구 검색(1 / 1)

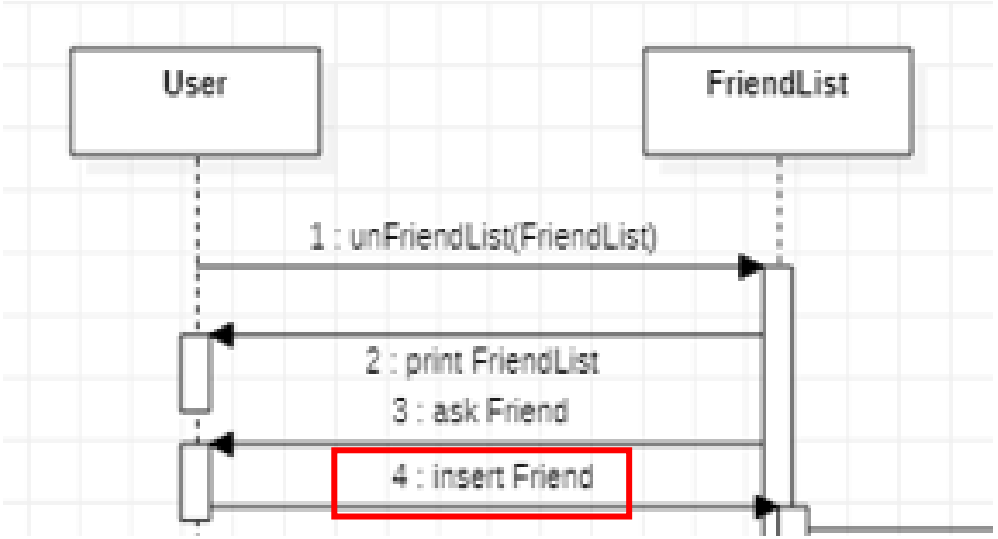
FriendList.java

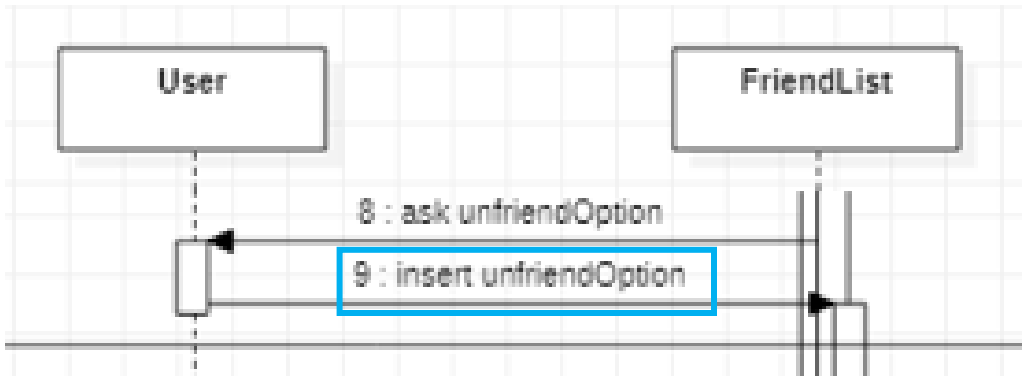



```
System.out.println("Insert strSearchWorld");      //U4.Sequence 2
String strSearchWorld = sc.next();                //U4.Sequence 3
/* U4.Sequence 3에서 Insert strSearchWorld 라는 부분은 메소드가 없는데 활성화를 가짐 수정 필요 */

boolean connect_result = kaKaoServer.connectServer(); //U4.Sequence 4, 5
/* U4.4: U1, U2 와 다르게 서버와 데이터베이스를 Create 로 생성 , 등일성 X */
if(connect_result == true){
```

# 03. 변경된 코드 – U5.친구 차단(1 / 2)

FriendList.java

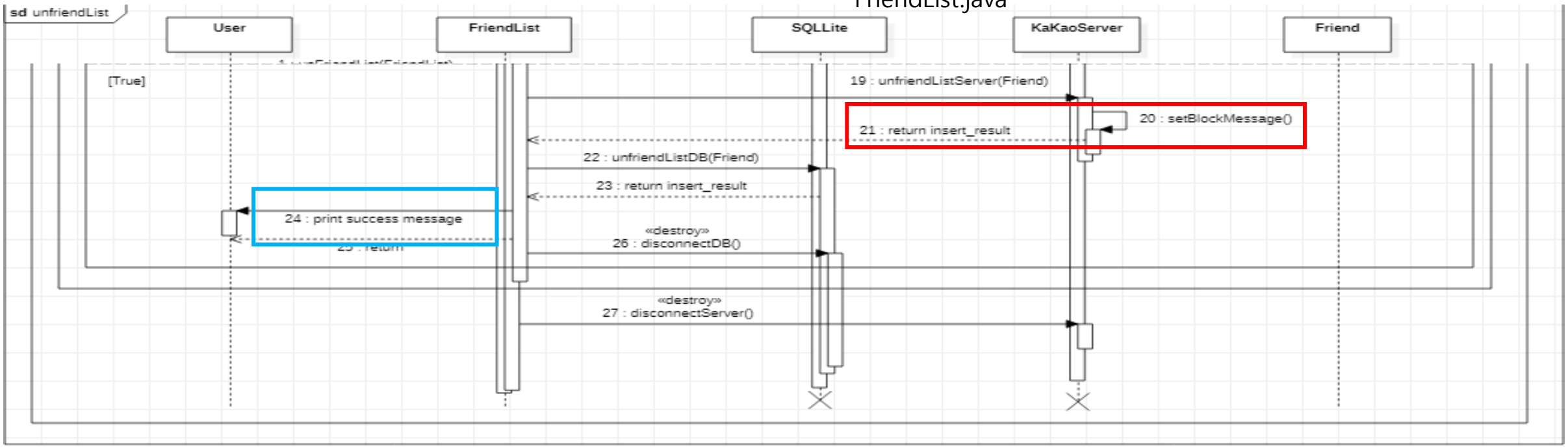



```
System.out.println("Insert Friend");          //U5.Sequence 3
int friend = sc.nextInt();                     //U5.Sequence 4
/* U5.4: InsertFriend 라는 메소드는 없는데 활성화를 가짐. */

boolean relation = FriendList.get(friend).IsMe();    //U5.Sequence 5, 6

if(relation == true){
    return;                                //U5.Sequence 7
}
System.out.println("Insert unfriendOption : Cancel, Agree");//U5.Sequence 8
String unfriendOption = sc.next();                          //U5.Sequence 9
/* U5.9: insert unfriendOption 라는 메소드는 없는데 활성화를 가짐. */
```

# 03. 변경된 코드 – U5.친구 차단(2 / 2)

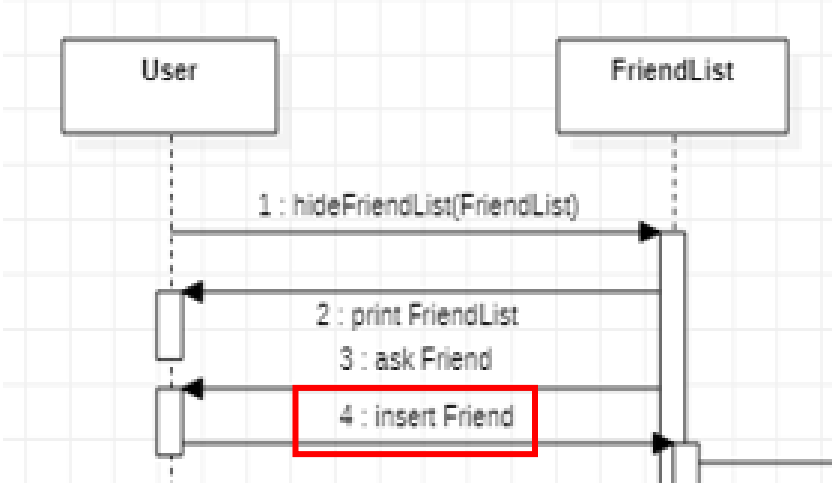
FriendList.java

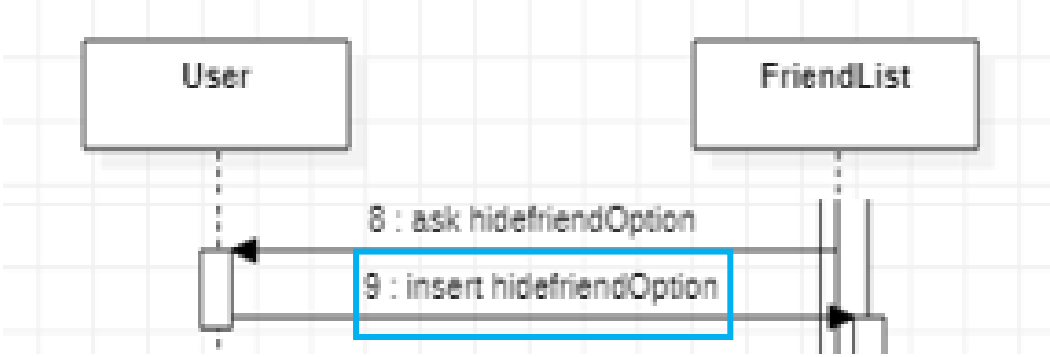



```
else{
    //U5.Sequence 19, 21
    boolean insert_result1 = kaKaoServer.unFriendListServer(FriendList.get(friend));
    /* U5.21: setBlockMessage() 메소드의 리턴 값은 void 이지만, insert_result1 을 리턴함. */

    //U5.Sequence 22, 23
    boolean insert_result2 = sqlLite.unFriendListDB(FriendList.get(friend));
    /* U5.24: 결과에 상관 없이 무조건 성공 메세지를 보냄. 아래와같이 if, else 로 수정함 */
    if(insert_result1 == true && insert_result2 == true){
        System.out.println("Success!");      //U5.Sequence24
        /* U5.25: 해당 부분을 실행하면 이후 26, 27 실행 불가, 따라서 삭제 필요 */
        /* return; */
    }
```

# 03. 변경된 코드 – U6.친구 숨김(1 / 3)

FriendList.java

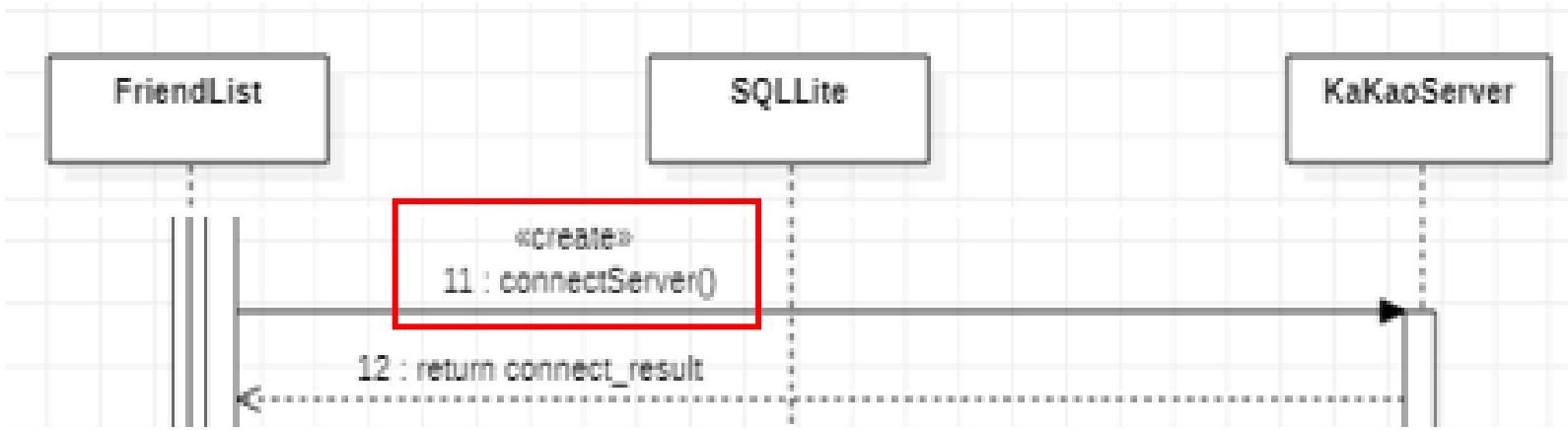



```
System.out.println("Insert Friend");                    //U6.Sequence 3
int friend = sc.nextInt();                               //U6.Sequence 4
/* U6.4: Insert Friend 라는 메소드는 없는데 활성화를 가짐. */

boolean relation = FriendList.get(friend).IsMe();        //U6.Sequence 5, 6
if(relation == true){
    return;                                              //U6.Sequence 7
}
System.out.println("Insert hideFriendOption");           //U6.Sequence 8
String hideFriendOption = sc.next();                     //U6.Sequence 9
/* U6.9: Insert hideFriendOption 라는 메소드는 없는데 활성화를 가짐. */
```

# 03. 변경된 코드 – U6.친구 숨김(2 / 3)

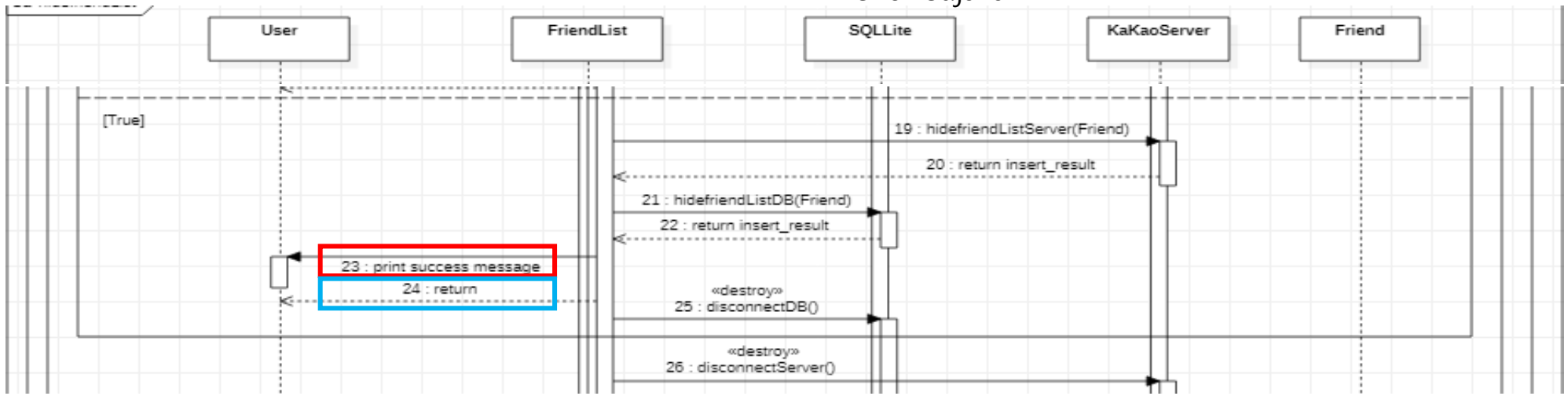
FriendList.java




```
else if(hideFriendOption.equals("Agree")){
    /* U6.11: U1, U2 와 다르게 서버와 데이터베이스를 Create 로 생성 , 통일성 X
    boolean connect_result = kaKaoServer.connectServer();
    if(connect_result == false) {
        System.out.println("Error!");                    //U6.Sequence 13
        return;
    }
```

# 03. 변경된 코드 – U6.친구 숨김(3 / 3)

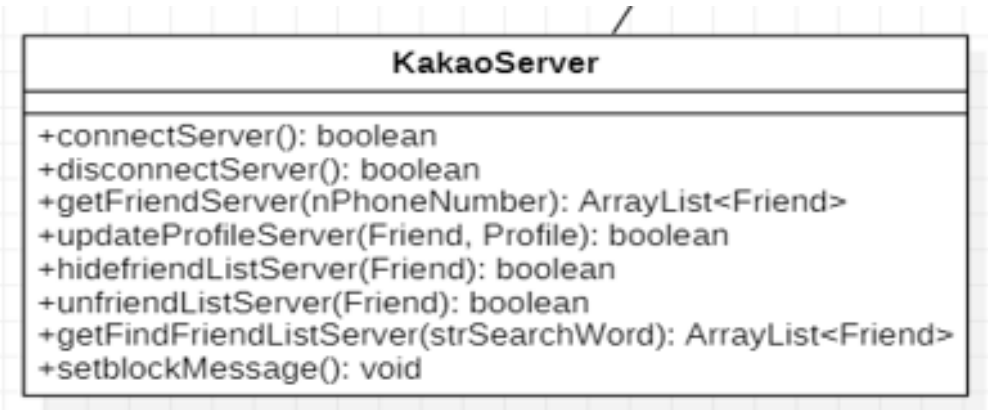
FriendList.java

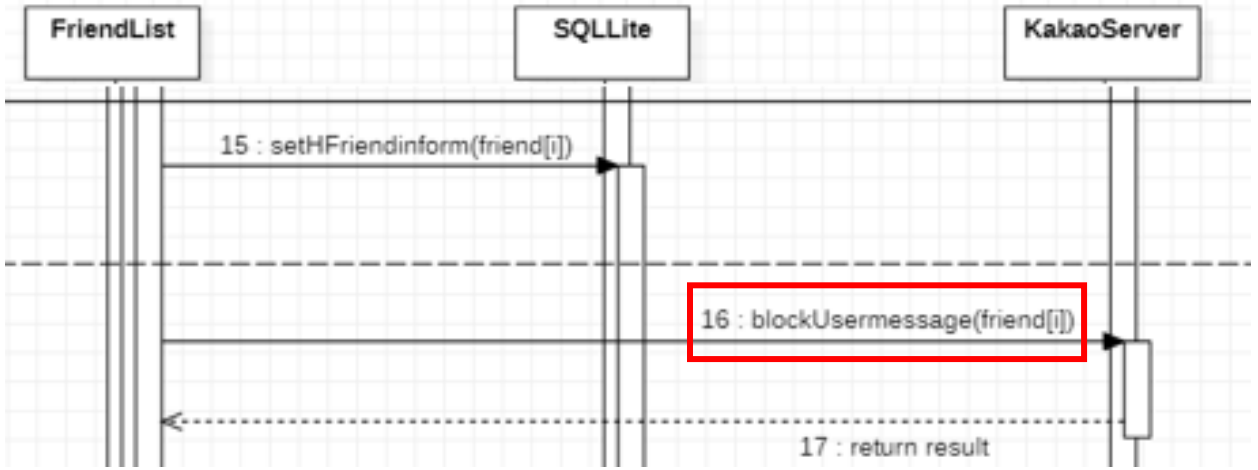



```
else {
    //U6.Sequence 19, 20
    boolean insert_result1 = kaKaoServer.hideFriendListServer(FriendList.get(friend));
    //U6.Sequence 21, 22
    boolean insert_result2 = sqlLite.hideFriendListDB(FriendList.get(friend));
    /* U6.23: 결과에 상관 없이 무조건 성공 메세지를 보냄. 아래와같이 if, else 로 수정함 */
    if(insert_result1 == true && insert_result2 == true){
        System.out.println("Success!");          //U6.Sequence 23
        /* U6.24: return 하게 되면, DB와 Server를 disconnect 못함. 따라서 return 부분 삭제*/
        /* return; */
    }
    else {
        System.out.println("Error!");
    }
    sqlLite.disconnectDB();          //U6.Sequence 25
}
kaKaoServer.disconnectServer();      //U6.Sequence 26
```

# 03. 변경된 코드 – U7.친구 목록 복구(1 / 3)

FriendList.java

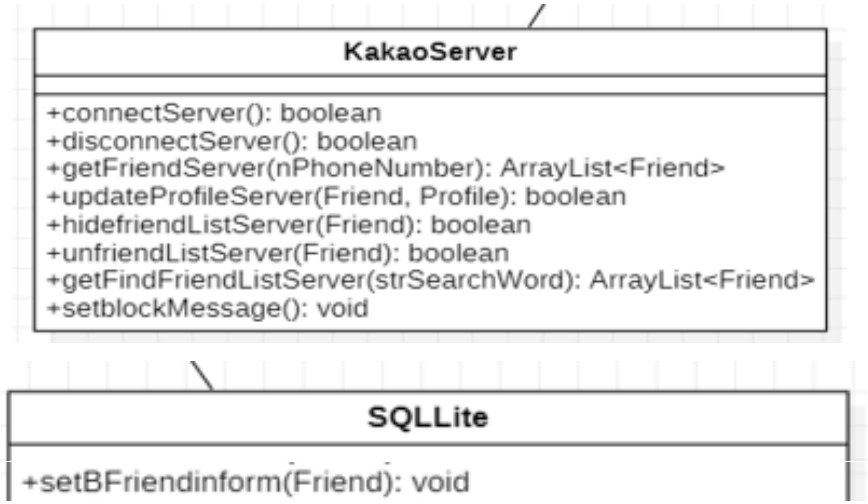


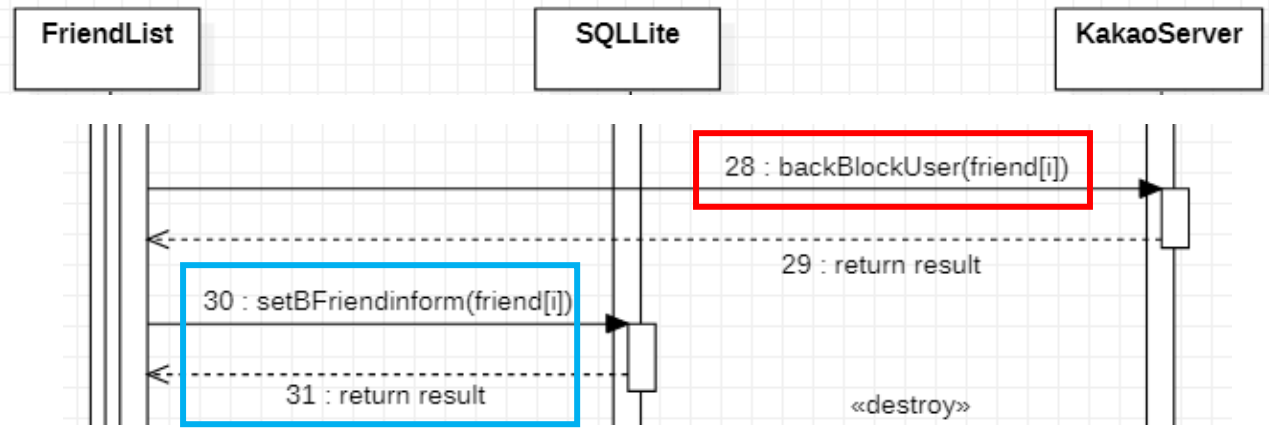




```
else if(option == 2){
    /* U7.18: blockUserMessage 메소드는 클래스다이어그램에 없음. */
    //U7.Sequence 18, 19
    boolean result = kaKaoServer.blockUserMessage(hideFriendList.get(friend));
    sqlLite.setBFriendInform(hideFriendList.get(friend));    //U7.Sequence 20
}
```

# 03. 변경된 코드 – U7.친구 목록 복구(2 / 3)

FriendList.java



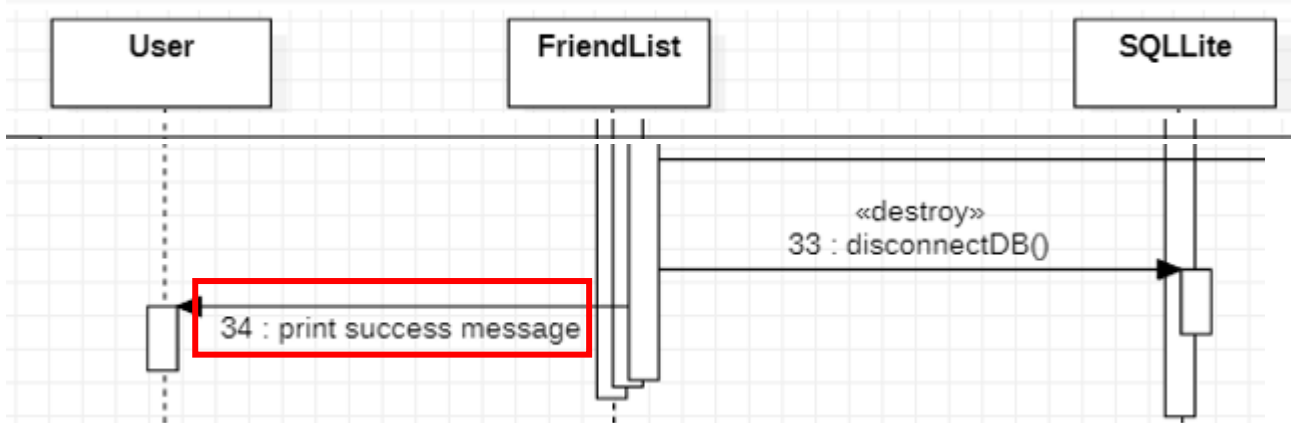




```java
//U7.Sequence 30, 31
/* U7.30: backBlockUser 메소드는 클래스다이어그램에 없음. */
boolean result1 = kaKaoServer.backBlockUser(unFriendList.get(friend));
//U7.Sequence 32, 33
/* U7.32: setBFriendInform 메소드 클래스다이어그램에서는 void, 통일성 X */
boolean result2 = sqlLite.setBFriendInform(unFriendList.get(friend));
```

# 03. 변경된 코드 – U7.친구 목록 복구(3 / 3)

FriendList.java

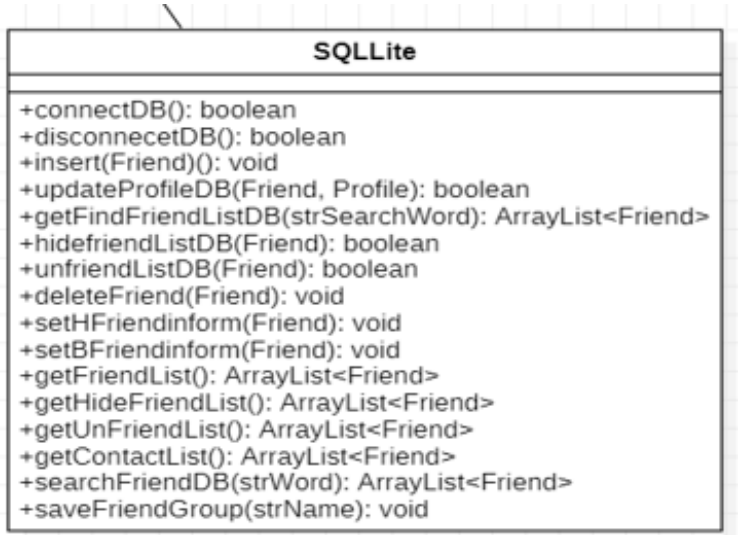
9. 선택된 ArrayList 친구를 매개변수로 setBFriendinform()를 호출한다. 해당 함수는 DB에 내장되어있다. boolean값을 리턴받고 서버와 DB연결을 해제한후 성공메세지를 출력한다. (리턴값이 FALSE인경우는 일단 제외)

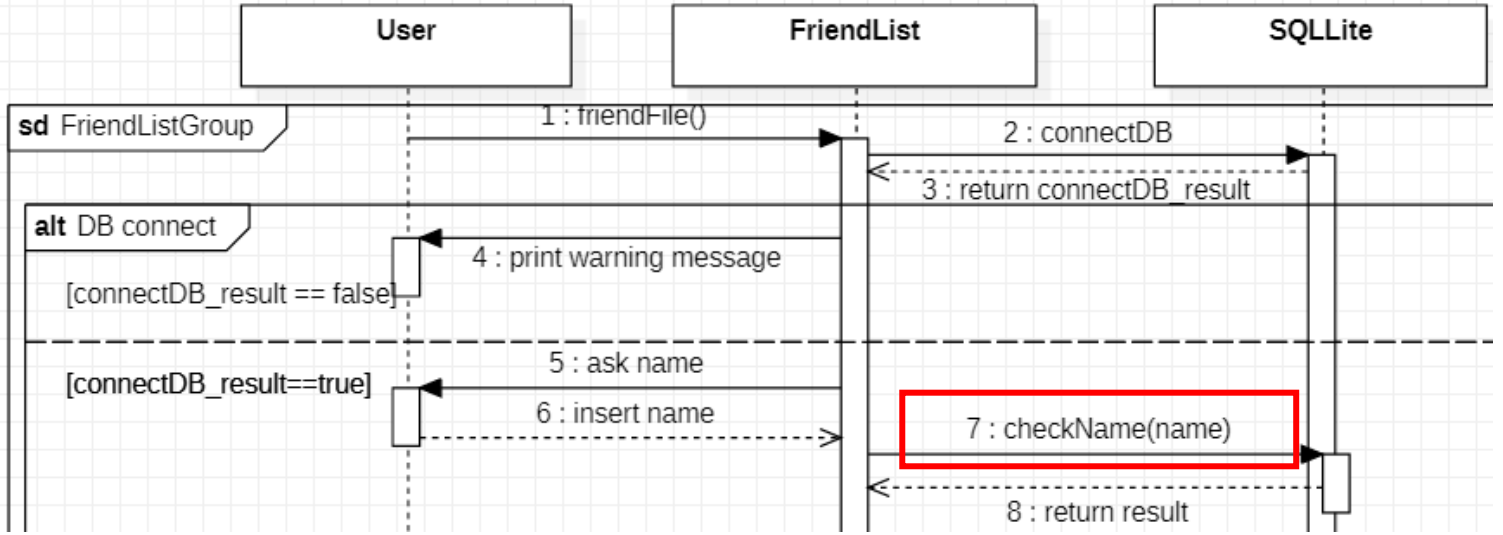



```java
/* U7.36: 설명은 result 값이 false 일때는 성공메세지를 보내지 않는다고 되어 있으나 */
/* 다이어그램에는 해당 내용 없음. 통일성 X, 설명대로 구현하였음. */
if(result1 == true && result2 == true){
    System.out.println("Success!");
}
```

# 03. 변경된 코드 – U8.친구 목록 그룹화(1 / 2)

FriendList.java




```
else{
    System.out.println("Insert name");   //U8.Sequence 5
    String name = sc.next();             //U8.Sequence 6

    /* U8.7: checkName 메소드는 클래스다이어그램에 없음. */
    boolean result = sqlLite.checkName(name);    //U8.Sequence 7, 8
    if(result == true){
```

# 03. 변경된 코드 – U8.친구 목록 그룹화(2 / 2)

FriendList.java


10. 생성된 FriendGroup(name)객체를 saveFriendGroup(객체이름)으로 저장한다. void형태

```
FriendGroup friendGroup = new FriendGroup(name);//U8.Sequence 10
```

⋮

```
/* U8.18: 친구 그룹을 추가 하는데, 이름만 등록함. friendGroup도 같이 저장해야 한다고 생각함.*/
/* 잘못된 부분: sqlLite.saveFriendGroup(name); */
sqlLite.saveFriendGroup(friendGroup, name);   //U8.Sequence 18
friendGroup = null;                           //U8.Sequence 19
sqlLite.disconnectDB();                       //U8.Sequence 20
```

# 04

## 어려웠던 부분

# 04. 어려웠던 부분

- 몇몇 시퀀스다이어그램의 설명에서 단순히 유저가 입력하는 부분을 표시한 부분들에 활성화 표기가 있음.
  (U3의 4번, 11번, 13번, U4의 3번, U5의 4번, 9번, U6의 4번, 9번)
  해당 부분들은 클래스다이어그램에 메소드로 존재하지 않는데 활성화가 되어 다른 메소드를 실행하거나 값을 리턴함.
  무엇을 원하는 다이어그램인지 정확하게 이해가 안됨.

- U5, U6에서 조건이 부합하지 않으면 return을 해서 종료시키는데, 몇몇 부분에서 종료하게 되면
  DB와 서버의 연결을 종료하지 못하게 구성되어있는 부분들이 있었음. 오류라고 판단하여 수정해서 구현하였음.

- U8 sequence 16은 입력받은 친구 이름은 String 이고, arraylist.get() 에서 필요한 변수는 int 라서 해당 부분을
  구현하기가 조금 어려웠음.

# 04. 어려웠던 부분

- 클래스다이어그램에서 각 메서드나 변수에 대한 설명이 없어, 이해하기 어려웠음

- 수업시간에 질문했던 U8 의 sequence 12 리턴 부분에 대한 시퀀스다이어그램 수정되지 않음.
- 수업시간에 질문했던 U8 의 alt check friend name 부분의 시퀀스다이어그램이 수정되지 않음.

# Q&A