P R O J E C T D O C U M E N T

System Model (Sequence Diagram) Document

제 2 조 NaLang

조원 : 201402387 이동원

201402330 김연훈

201604136 김노은

지도교수: OOO (서명)

Document Revision History

REV#	DATE	AFFECTED SECTION	AUTHOR
1	2020/05/15	3.1~3.3	이동원
2	2020/05/15	3.4~3.6	김노은
3	2020/05/15	3.7~3.9	김연훈

Table of Contents

1. INTRODUCTION 5

1.1. OBJECTIVE 5

2. USE CASE DIAGRAM 6

3. SEQUENCE DIAGRAM 7

3.1. AMSM_REQ_MONITORING_N001 (SUBSCRIBEESESTATUS) 7

List of Figure

FIGURE 1 – USE CASE DIAGRAM 6
FIGURE 2 – ESE STARTUP SEQUENCE DIAGRAM 7

1. Introduction

1.1. Objective

이 문서는 자연어 처리 기술을 이용한 일기 어플리케이션의 시스템 모델의(시퀀스 다이어그램)에 대한 내용을 기술하고 있다. 요구사항 명세 단계에서 작성한 유스케이스 다이어그램을 기반으로 각 유스케이스의 상세한 내부 동작 흐름을 시퀀스 다이어그램으로 모델링한다.

2. Use Case Diagram

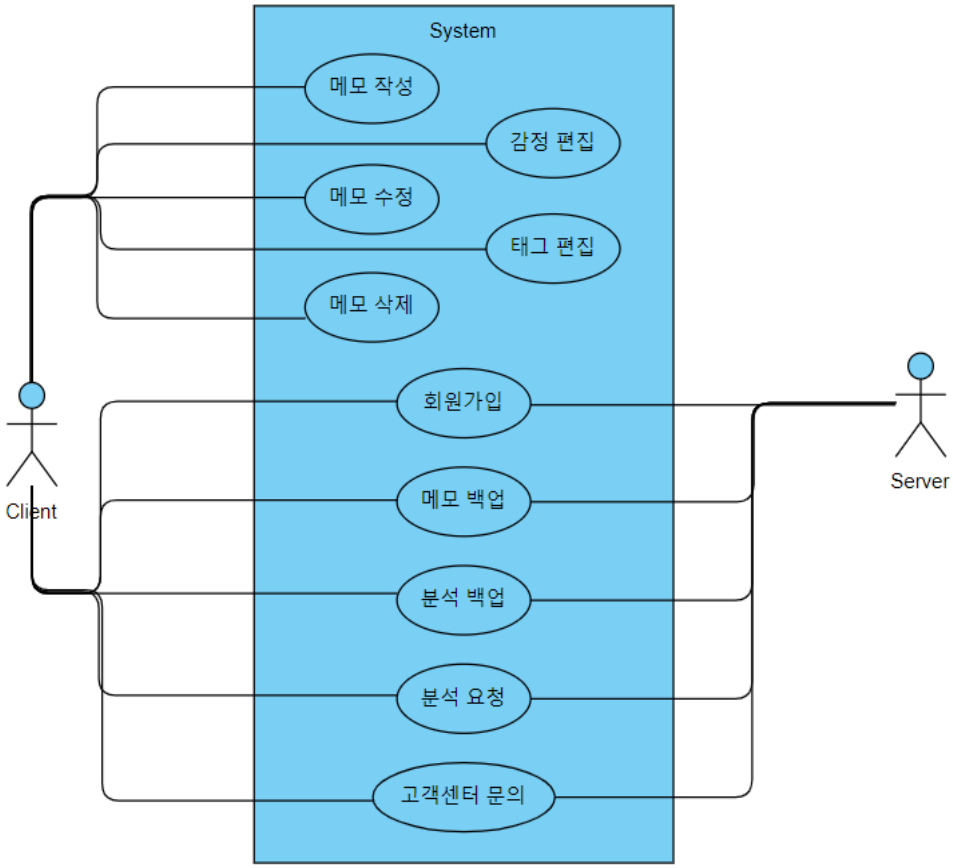


Figure 1 – Use Case Diagram

3. Sequence Diagram

3.1. 회원가입

USE_01은 Client의 요청에 따라 회원 정보를 서버에 등록하는 동작에 대한 요구사항이다.

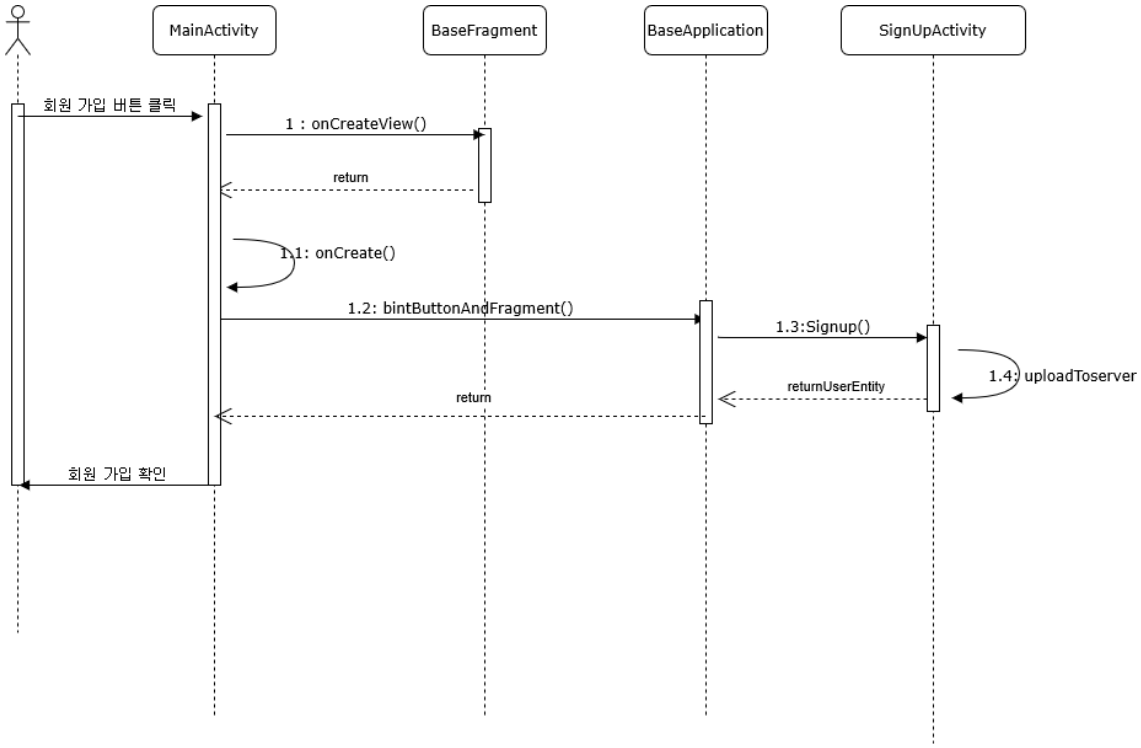


Figure 1 – SignUp Sequence Diagram

1. 사용자는 회원가입 버튼을 클릭하고 MainActivity는 onCreateView 메소드를 통해 BaseFragment 에서 화면 xml을 요청해 받는다.

2. 사용자는 onCreate()메소드를 통해 구성된 화면을 확인한다.

3. MainActivity는 bintButtonAndFragment()메소드를 통해 사용자가 누른 버튼의 동작을 수행한다.

   3.1: BaseApplication에서 SignUpActivity를 실행시키는 Singup()메소드를 실행시킨다.

   3.2: SignUpActivity는 uploadToServer()메소드로 사용자 정보를 서버로 업로드한다.

   3.3: 회원가입이 이루어 졌음을 사용자에게 확인시킨다.

3.2. 메모작성

USE_02은 Client의 요청에 따라 메모 데이터를 서버에 등록하는 동작에 대한 요구사항이다.

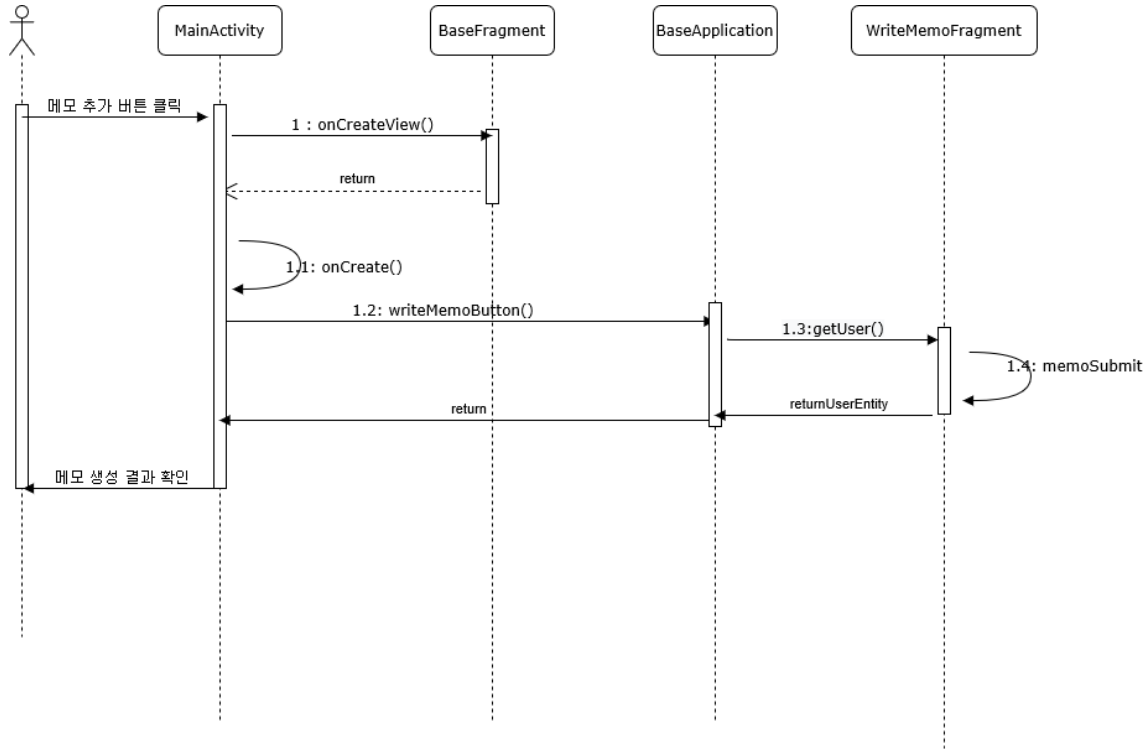


Figure 2 – WriteMemoSequence Diagram

1. 사용자는 메모 작성 버튼을 클릭하고 MainActivity는 onCreateView 메소드를 통해 BaseFragment 에서 화면 xml을 요청해 받는다.

2. 사용자는 onCreate()메소드를 통해 구성된 화면을 확인한다.

3. MainActivity는 writeMemoButton()메소드를 통해 사용자가 누른 버튼의 동작을 수행한다.

   3.1: BaseApplication에서 getUser()메소드를 통해 해당하는 사용자의 정보로 서버에 접근한다.

   3.2: 특정 UserEntity에서 WriteMemoFragment로 접근해 편집을 시도한다.

   3.3: WriteMemoFragment는 memoSubmit()메소드를 통해 메모를 등록한다.

   3.4: 등록된 메모를 반환하여 사용자에게 확인시킨다.

3.3. 메모수정

USE_03은 Client의 요청에 따라 메모 데이터를 수정하여 서버에 반영하는 동작에 대한 요구사항이다.

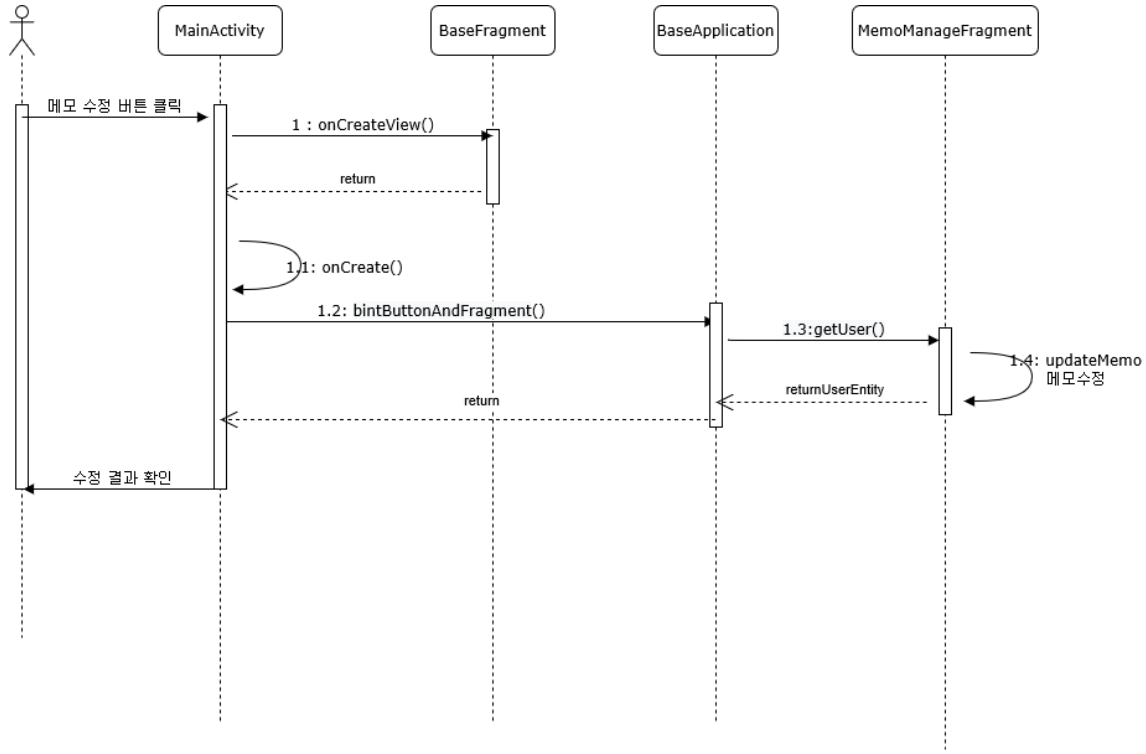


Figure 3 – UpdateMemoSequence Diagram

1. 사용자는 메모 작성 버튼을 클릭하고 MainActivity는 onCreateView 메소드를 통해 BaseFragment에서 화면 xml을 요청해 받는다.

2. 사용자는 onCreate()메소드를 통해 구성된 화면을 확인한다.

3. MainActivity는 binButtonAndFragment()메소드를 통해 사용자가 누른 버튼의 동작을 수행한다.

   3.1: BaseApplication에서 getUser()메소드를 통해 해당하는 사용자의 정보로 서버에 접근한다.

   3.2: 특정 UserEntity에서 MemoManageFragment로 접근해 편집을 시도한다.

   3.3: WriteMemoFragment는 uupdateMemo()메소드를 통해 메모를 갱신한다.

   3.4: 갱신된 메모를 반환하여 사용자에게 확인시킨다.

3.4. 메모 삭제

USE_04는 Client의 메모 삭제 요청에 따라 메모 데이터를 수정하여 서버에 반영하는 동작에 대한 요구사항이다.

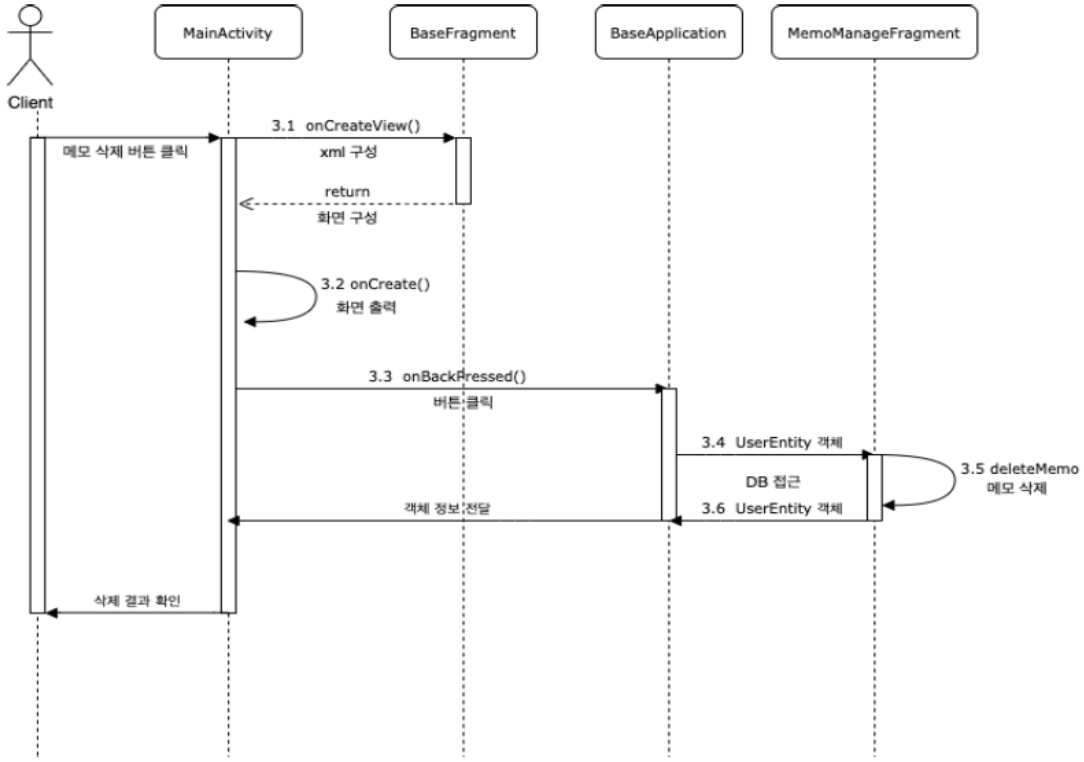


Figure 4 – Delete Memos Sequence Diagram

1. 사용자는 메모 백업 버튼을 클릭하고 MainActivity는 onCreateView 메소드를 통해 BaseFragment에서 화면 xml 규격을 요청해 받는다.

2. 사용자는 onCreate() 메소드를 통해 구성된 화면을 확인한다.

3. MainActivity는 onBackPressed() 메소드를 통해 사용자가 누른 버튼의 동작을 수행한다.

3-1. BaseApplication에서 UserEntity를 통해 해당하는 사용자의 정보로 서버에 접근한다.

3.2. 특정 UserEntity에서MemoManageFragment로 접근해 태그 편집을 시도한다.

3.3. MemoManageFragments는 deleteMemo() 메소드를 통해 메모를 삭제한다.

3.4. 삭제된 메모 정보를 반환하여 사용자에게 수정 결과를 확인 시킨다.

3.5. 태그 편집

USE_05은 Client의 태그 편집 요청에 따라 태그 데이터를 수정하여 서버에 반영하는 동작에 대한 요구사항이다.

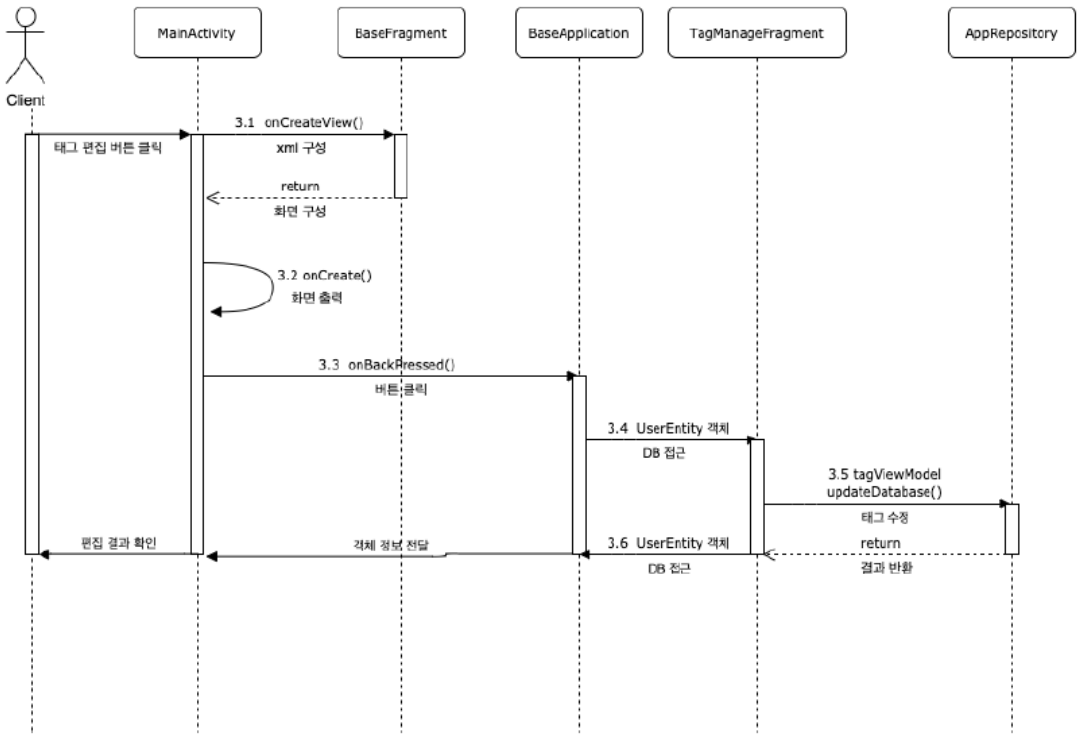


Figure 5 – Edit Tags Sequence Diagram

1. 사용자는 메모 백업 버튼을 클릭하고 MainActivity는 onCreateView 메소드를 통해 BaseFragment에서 화면 xml 규격을 요청해 받는다.

2. 사용자는 onCreate() 메소드를 통해 구성된 화면을 확인한다.

3. MainActivity는 onBackPressed() 메소드를 통해 사용자가 누른 버튼의 동작을 수행한다.

3-1. BaseApplication에서 UserEntity를 통해 해당하는 사용자의 정보로 서버에 접근한다.

3.2. 특정 UserEntity에서 TagManageFragment로 접근해 태그 편집을 시도한다.

3.3. TagManageFragment는태그 데이터 수정을 위해 tagViewModel을 통해 AppRepository에 접근하여
태그 수정을 시도한다.

3.4. AppRepository를 통해 수정된 태그 정보를 반환하여 사용자에게 수정 결과를 확인 시킨다.

3.6. 메모 백업

USE_06은 Client의 메모 백업 요청에 따라 초컬 메모 데이터를 서버 측의 메모 데이터로 덮어씌우는 동작에 대한 요구사항이다.

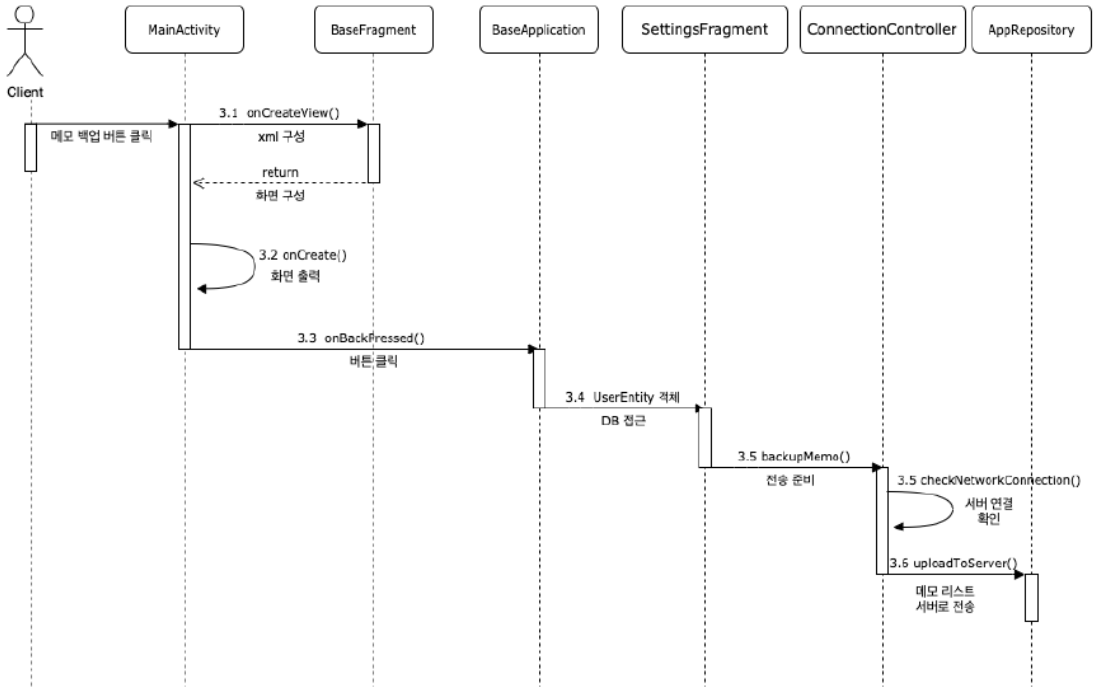
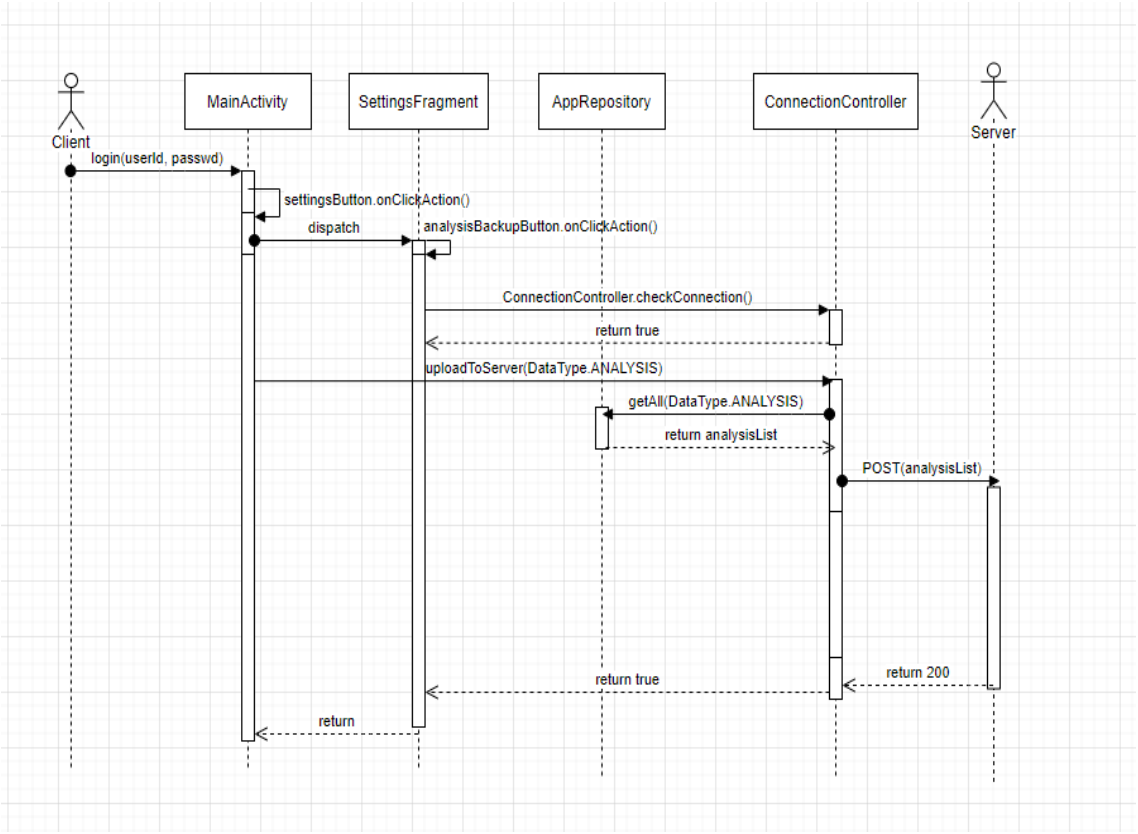


Figure 6– Backup Memos Sequence Diagram

1. 사용자는 메모 백업 버튼을 클릭하고 MainActivity는 onCreateView 메소드를 통해 BaseFragment에서 화면 xml 규격을 요청해 받는다.
2. 사용자는 onCreate() 메소드를 통해 구성된 화면을 확인한다.
3. MainActivity는 onBackPressed() 메소드를 통해 사용자가 누른 버튼의 동작을 수행한다.
4. BaseApplication에서 UserEntity를 통해 해당하는 사용자의 정보로 서버에 접근한다.
5. 특정 UserEntity에서 SettingFragment를 통해 backupMemo() 메소드를 실행한다.
6. backupMemo() 메소드는 ConnectionController에 접근하여 checkNetworkConnection() 메소드로 연결 상태를 확인 후 uploadToServer() 메소드를 통해 메모 데이터 백업을 수행한다.

3.7. 분석 백업



1. 사용자는 사용자의 아이디와 비밀번호(userId, passwd)로 로그인한다.

2. 메인 화면의 메뉴에서 설정 버튼(settingsButton)을 클릭하여 설정 화면으로 전환한다.

3. 설정 화면에서 분석 백업 버튼(analysisBackupButton)을 클릭하여 분석 백업을 실행한다.

    3.1: ConnectionController를 통해 서버와 통신이 가능한지 확인한다.

    3.2: 통신이 가능하면 ConnectionController를 통해 서버에 분석 데이터 목록을 업로드한다.

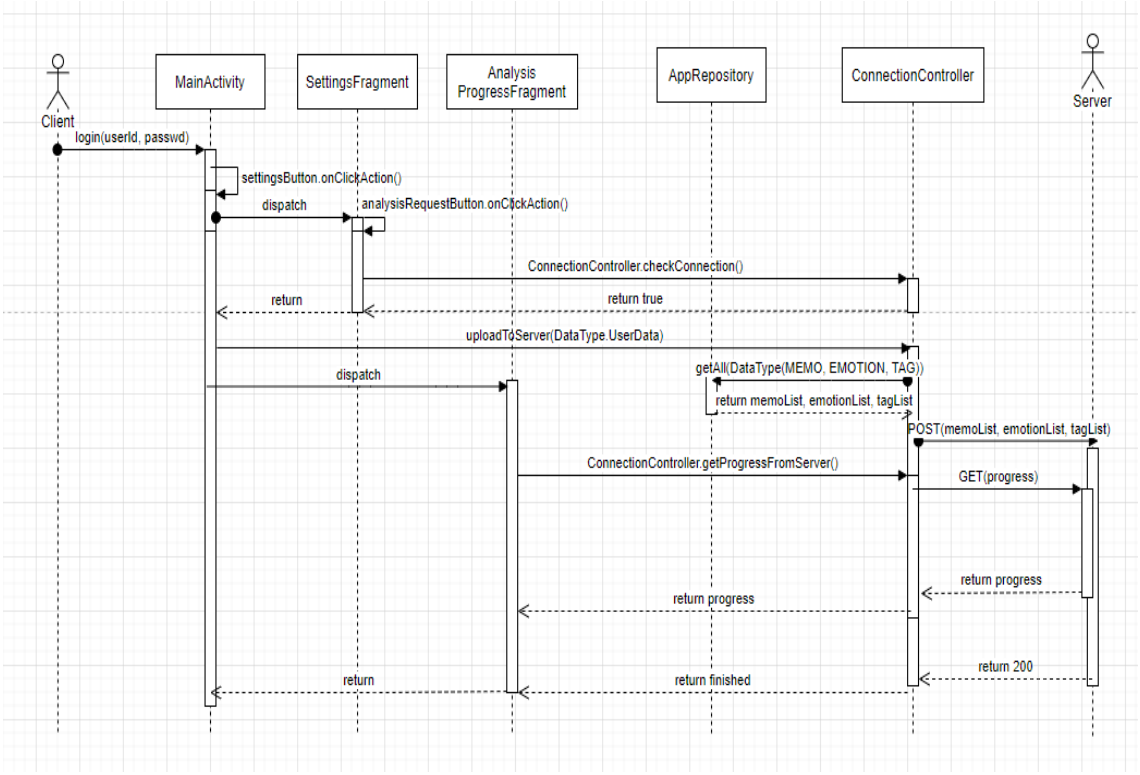
        3.2.1: ConnectionController는 AppRepository로부터 분석 데이터 목록을 불러온다.

        3.2.2: ConnectionController는 POST 방식으로 분석 데이터 목록을 전송한다.

    3.3 전송이 완료되어 서버로부터 응답 코드 200을 받는다.

4. 설정 화면에서 백업 완료 토스트 메세지를 출력한다.

3.8. 분석 요청



1. 사용자는 사용자의 아이디와 비밀번호(userId, passwd)로 로그인한다.

2. 메인 화면의 메뉴에서 설정 버튼(settingsButton)을 클릭하여 설정 화면으로 전환한다.

3. 설정 화면에서 분석 백업 버튼(analysisBackupButton)을 클릭하여 분석 요청을 실행한다.

    3.1: ConnectionController를 통해 서버와 통신이 가능한지 확인한다.

        3.1.1: 통신이 가능한 경우 분석 과정 출력 화면으로 전환한다.

    3.2: 통신이 가능하면 ConnectionController를 통해 서버에 사용자 데이터 목록을 업로드한다.

        3.2.1: ConnectionController는 AppRepository로부터 사용자 데이터 목록을 불러온다.

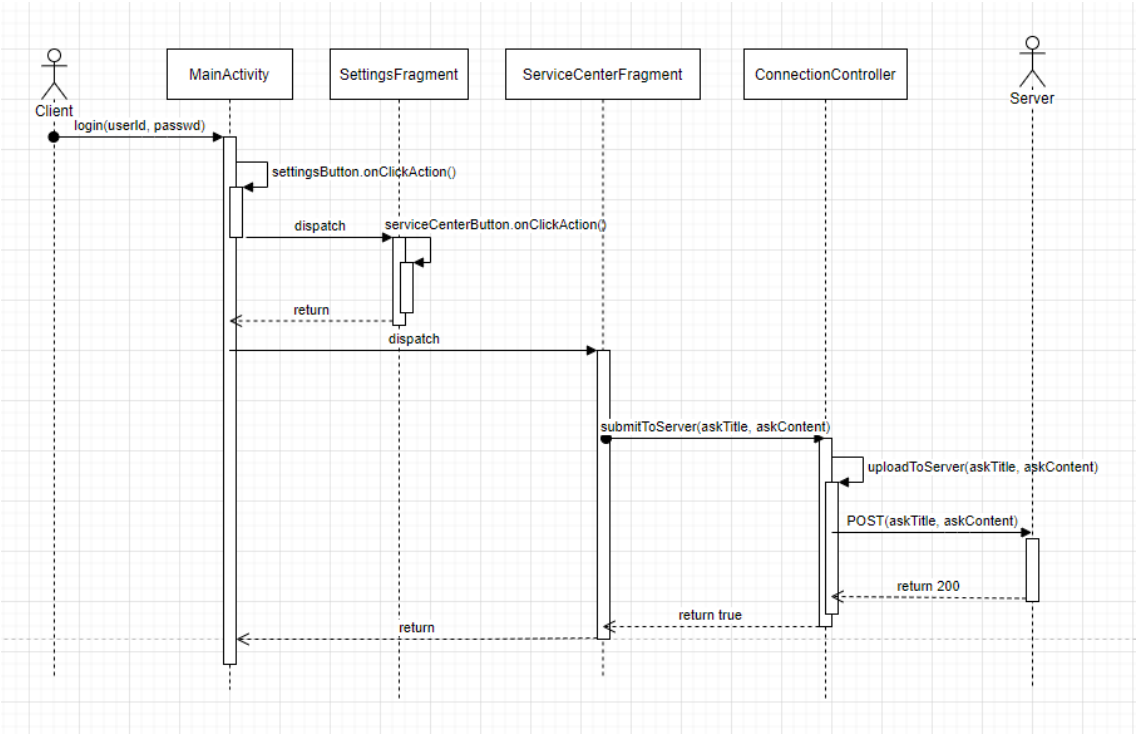
        3.2.2: ConnectionController는 POST 방식으로 사용자 데이터 목록을 전송한다.

    3.3: 전송 중에 분석 과정 출력 페이지를 보는 경우 서버로부터 진행도를 받아와 확인할 수 있다.

    3.4: 전송이 완료되어 서버로부터 응답 코드 200을 받는다.

4. 분석 과정 출력 화면에서 분석이 완료되었음을 확인할 수 있다.

3.9. 고객센터 문의



1. 사용자는 사용자의 아이디와 비밀번호(userId, passwd)로 로그인한다.

2. 메인 화면의 메뉴에서 설정 버튼(settingsButton)을 클릭하여 설정 화면으로 전환한다.

3. 설정 화면에서 고객 센터 버튼(serviceCenterButton)을 클릭하여 고객 센터 화면으로 전환한다.

4. 고객 센터 화면에서 문의 제목과 문의 내용을 작성하여 서버에 제출한다.

    4.1: ConnectionController를 통해 서버와 통신이 가능한지 확인한다.

    4.2: 통신이 가능하면 ConnectionController를 통해 서버에 문의 제목과 문의 내용을 업로드한다.

    4.3: 전송이 완료되어 서버로부터 응답 코드 200을 받는다.

5. 고객 센터 화면에서 문의 제출 완료 토스트 메세지를 출력한다.

유튜브 링크 : https://youtu.be/gfrYjcC_WA8

깃허브링크: https://github.com/yeonhunkim/Reminder/tree/master/software_engineering/sequence