요약

소프트웨어 공학 시간에 공부한 내용을 바탕으로  
요구사항 분석 보고서를 작성하였다.

Final Report

Software engineering

하루끝

용김강허여

목 차

[1. Project Overview 4](#_Toc73990187)

[1.1. Problem Statement and Motivation 4](#_Toc73990188)

[1.2. Proposed Idea and Solution 5](#_Toc73990189)

[2. Scope and Objective 5](#_Toc73990190)

[3. User Requirements 6](#_Toc73990191)

[4. System Requirements 7](#_Toc73990192)

[4.1. Functional Requirements 7](#_Toc73990193)

[4.2. Non-Functional Requirements 8](#_Toc73990194)

[5. Requirements Specification 9](#_Toc73990195)

[5.1. Usecase 1 - 선택 작성 9](#_Toc73990196)

[5.2. Usecase 2 - 직접 작성 9](#_Toc73990197)

[5.3. Usecase 3 - Todo List 10](#_Toc73990198)

[5.4. Usecase 4 - 교환일기 10](#_Toc73990199)

[5.5. Usecase 5 - Analysis and Visualization 11](#_Toc73990200)

[6. Requirements Analysis 12](#_Toc73990201)

[6.1. Behavioral Analysis 12](#_Toc73990202)

[6.1.1. Usecase 1 - 선택 작성 12](#_Toc73990203)

[6.1.2. Usecase 2 - 직접 작성 13](#_Toc73990204)

[6.1.3. Usecase 3 - Todo List 13](#_Toc73990205)

[6.1.4. Usecase 4 - 교환 일기 14](#_Toc73990206)

[6.1.5. Usecase 5 - Analysis and Visualization 14](#_Toc73990207)

[6.2. Structural Analysis 15](#_Toc73990208)

[6.2.1 Briefly describe each analysis class 15](#_Toc73990209)

[7. Architectural Design 17](#_Toc73990210)

[7.1. Logical View / Architecture 17](#_Toc73990211)

[7.1.1. Layer Description 17](#_Toc73990212)

[8. Design Modeling 18](#_Toc73990213)

[8.1. Usecase 1 - 선택 작성 18](#_Toc73990214)

[8.1.1. Design Sequence Diagram 18](#_Toc73990215)

[8.1.1.1. <scenario 1> 선택 작성 18](#_Toc73990216)

[8.1.2. Design Class Diagram 19](#_Toc73990217)

[8.2. Usecase 2 - 직접 작성 19](#_Toc73990218)

[8.2.1. Design Sequence Diagram 19](#_Toc73990219)

[8.2.1.1. <scenario 2> 직접 작성 19](#_Toc73990220)

[8.2.2. Design Class Diagram 20](#_Toc73990221)

[8.3. Usecase 3 - Todo List 20](#_Toc73990222)

[8.3.1. Design Sequence Diagram 20](#_Toc73990223)

[8.3.1.1. <scenario 3-1> Todo List 추가 20](#_Toc73990224)

[8.3.1.2. <scenario 3-2> Todo List 삭제 21](#_Toc73990225)

[8.3.1.3. <scenario 3-2> Todo List 수정 21](#_Toc73990226)

[8.3.2. Design Class Diagram 21](#_Toc73990227)

[8.4. Usecase 4 - 교환 일기 22](#_Toc73990228)

[8.4.1. Design Sequence Diagram 22](#_Toc73990229)

[8.4.1.1 <scenario 4> 교환 일기 22](#_Toc73990230)

[8.4.2. Design Class Diagram 23](#_Toc73990231)

[8.5. Usecase 5 - Analysis and Visualization 24](#_Toc73990232)

[8.5.1. Design Sequence Diagram 24](#_Toc73990233)

[8.5.1.1 <scenario 5> 사용자 기록 시각화 24](#_Toc73990234)

[8.5.2 Design Class Diagram 25](#_Toc73990235)

[8.6. Combined Design Class Diagram 25](#_Toc73990236)

[9. Implementation 26](#_Toc73990237)

[9.1. Implementation Specific 26](#_Toc73990238)

[9.1.1. end\_of\_day 26](#_Toc73990239)

[9.1.1.1. settings.py 26](#_Toc73990240)

[9.1.1.2. urls.py 26](#_Toc73990241)

[9.1.2. accountapp 27](#_Toc73990242)

[9.1.2.1. views.py 27](#_Toc73990243)

[9.1.2.2. urls.py 27](#_Toc73990244)

[9.1.2.3. models.py 27](#_Toc73990245)

[9.1.2.4. forms.py 27](#_Toc73990246)

[9.1.3. profileapp 27](#_Toc73990247)

[9.1.3.1. views.py 27](#_Toc73990248)

[9.1.3.2. urls.py 27](#_Toc73990249)

[9.1.3.3. models.py 27](#_Toc73990250)

[9.1.3.4. forms.py 27](#_Toc73990251)

[9.1.4. diaryapp 28](#_Toc73990252)

[9.1.4.1. views.py 28](#_Toc73990253)

[9.1.4.2. urls.py 28](#_Toc73990254)

[9.1.4.3. models.py 28](#_Toc73990255)

[9.1.4.4. forms.py 28](#_Toc73990256)

[9.1.5. crossdiaryapp 28](#_Toc73990257)

[9.1.5.1. views.py 28](#_Toc73990258)

[9.1.5.2. urls.py 28](#_Toc73990259)

[9.1.5.3. models.py 28](#_Toc73990260)

[9.1.6. static 28](#_Toc73990261)

[9.1.7. media 29](#_Toc73990262)

[9.1.8. templates 29](#_Toc73990263)

[9.1.9 db.sqlite3 29](#_Toc73990264)

[9.2. Message Format (or any other contents) 29](#_Toc73990265)

[9.3. System/Data Dependencies & Requirements 29](#_Toc73990266)

[10. UI Design 30](#_Toc73990267)

[10.1. account 30](#_Toc73990268)

[10.2. profile 31](#_Toc73990269)

[10.3. diary 32](#_Toc73990270)

[10.4. cross diary 33](#_Toc73990271)

[11. Test Plan & Result 34](#_Toc73990272)

[11.1. Validation criteria & Scope 34](#_Toc73990273)

[11.1.1 다이어리 선택작성 34](#_Toc73990274)

[11.1.2. 다이어리 직접작성 34](#_Toc73990275)

[11.1.3. to-do list 34](#_Toc73990276)

[11.1.4. 교환일기 34](#_Toc73990277)

[11.1.5. 시각화 34](#_Toc73990278)

[11.2. Test Plan 35](#_Toc73990279)

[11.2.1. Software to be Tested 35](#_Toc73990280)

[11.2.2. Testing Strategy 35](#_Toc73990281)

[11.2.2.1. Unit testing 35](#_Toc73990282)

[11.2.2.2. Integration / System testing 35](#_Toc73990283)

[11.2.3. Testing Tools and Environment 35](#_Toc73990284)

[11.3. Integration/System Tests 36](#_Toc73990285)

[11.3.1. 다이어리 선택작성 36](#_Toc73990286)

[11.4.2. 다이어리 직접작성 36](#_Toc73990287)

[11.4.3. todo list 37](#_Toc73990288)

[11.4.4. 교환일기 37](#_Toc73990289)

[11.4.5. 시각화 39](#_Toc73990290)

[11.5. Test Results 40](#_Toc73990291)

[11.6. Requirements Satisfaction 40](#_Toc73990292)

[11.6.1. User Requirements 40](#_Toc73990293)

[11.6.2. Non-Functional Requirement 41](#_Toc73990294)

[Appendix 42](#_Toc73990295)

[A. Glossary 42](#_Toc73990296)

[B. Domain class Description 42](#_Toc73990297)

[C. Project Status 46](#_Toc73990298)

[C.1 Planned vs. actual 46](#_Toc73990299)

[C.2 Team members’ contribution 46](#_Toc73990300)

[E. Change History 47](#_Toc73990301)

[F. References 48](#_Toc73990302)

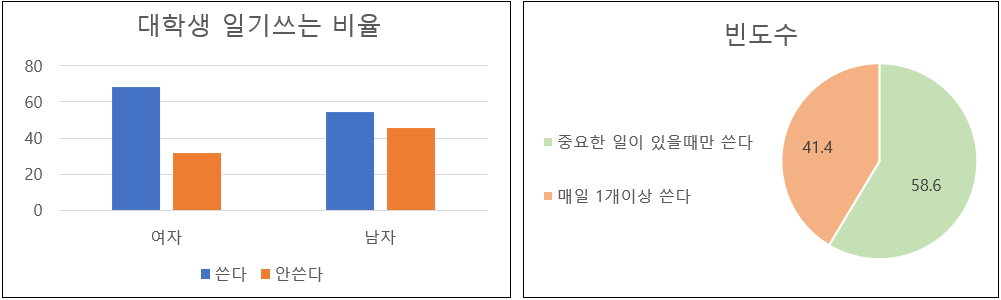
# 

# 1. Project Overview

## 1.1. Problem Statement and Motivation

일기를 쓰는 사람들이 많지 않은 사실을 해결하고자 이 프로젝트를 시작하였다. 실제로 우리는 일기를 쓰는 것이 정서적으로 안정감을 준다는 사실을 알고 있음에도 일기를 쓰는 데 조금의 시간도 소비하지 않는다. 그림 1을 보면 여학생의 경우, 일기 쓰는 비율이 꽤 높지만 남학생은 일기를 쓰지 않는 비율이 높다. 또한 그림 2를 보면 일기 쓰는 학생들은 매일 쓰기보다는 중요한 일이 있을 때 일기를 쓰는 빈도가 높다는 것을 알 수 있다.[[1]](#footnote-1) 이 자료를 통해서 일기를 쓰지 않는 가장 큰 원인을 매일 ‘글’ 쓰는 것이 귀찮기 때문이라고 판단하였다.

그렇다면 왜 일기를 써야 할까? 정신과 의사들이 일기를 쓰도록 권장하는 가장 큰 이유는 나의 기분을 글로 남기면 너무 up되어 있는 감정은 내리고, 너무 down되어 있는 감정은 올림으로써 감정적으로 안정적일 수 있게 해주기 때문이다. 일기는 하루를 돌아보고 정리하고, 나의 삶의 기록을 만들 수도 있다. 그리고 과거의 자신을 되돌아보며 반성하고 발전할 수 있는 계기를 마련할 수 있다. 이러한 순기능을 가진 일기를 많은 사람들이 쉽고 편리하게 쓰도록 하기 위해 본 프로젝트를 진행하고자 한다.



**<그림 1. 대학생 일기 쓰는 비율>** **<그림 2. 일기 쓰는 빈도>**

## 

## 1.2. Proposed Idea and Solution

대부분의 사람들은 일기를 쓰면 얻게 되는 좋은 점들을 정확하게 인지하고 있다. 하지만 일기를 매일 쓰는 것이 상당히 힘들고 꾸준히 씀으로써 느끼는 성취감을 얻기 또한 힘들다. 따라서 본 프로젝트는 일기를 쉽고 편하게 쓸 수 있는 방법을 제시해주고 꾸준히 썼을 때 성취감을 느낄 수 있도록 해준다. 그렇게 일기를 꾸준히 쓰게 된다면 하루동안 특별한 일이 없더라도 자신의 생각과 감정을 정리할 수 있게 될 것이다. 또한, 일기와 더불어 계획적인 삶을 만들어 줄 수 있는 기능을 제공하면 효율적인 일상생활을 보내도록 도와줄 것이다. 이는 코로나19가 지속되는 상황 속에서 우울증을 극복하는데도 큰 도움이 된다.[[2]](#footnote-2)

다음은 우리가 제안하는 solution이다.

일단, 사용자가 일기를 꾸준히 쓰게 하기 위해서 일기를 쉽게 쓰게 하기 위해 일기를 처음부터 끝까지 전문을 쓰는 직접작성 방식과 사건을 육하원칙의 질문에 답변하는 방식을 제공한다. 그리고 꾸준히 일기를 쓸 수 있는 동기를 만들어주기 위해 교환일기 기능을 제공한다. 혼자서 일기를 작성하는 것이 아니라 같이 작성하는 파트너를 구해 일기를 나누어, 작성하는 의무감을 자극하는 것이다. 또한, 텍스트 분석과 일기 작성 횟수 등을 시각적으로 보여줘 일기를 썼을 때 결과를 쉽게 눈으로 확인할 수 있게 해주고, To-do list 기능을 추가하여 해야 할 일들을 정리하고, 실천했을 때 기록이 되도록 한다.

# 2. Scope and Objective

|  |  |
| --- | --- |
| In the scope | Out of scope |
| - 일기 직접 작성 페이지 추가  - 일기 선택 작성 페이지 추가  - 웹에서 구현 (반응형 웹으로 구현)  - 회원가입을 통한 privacy 보장  - 아이디, 비밀번호 찾기는 가입할 때  이메일을 통해서 찾기  - 텍스트 분석을 통한 주기 별 단어 통계  - 교환일기는 사용자 모두 작성권한 제공  - todo list 구현 | - 한국어만 지원  - 모바일 앱으로 개발하지 않음  - 교환일기 자체를 사용자가 삭제하는 기능은  제공하지 않음  - 휴대폰 번호를 활용한 인증 절차는  개발하지 않음 |

이번 프로젝트의 궁극적인 목표는 사용자가 하루를 마무리하기 위한 일기를 쉽고 재밌게 꾸준히 작성할 수 있도록 다양한 기능을 제공해 성취감과 일기 쓰기를 취미로 삼을 수 있도록 새로운 경험을 제공하는 웹 다이어리 서비스를 개발하는 것이다.

# 3. User Requirements

|  |  |
| --- | --- |
| No. | USER\_RS\_001 |
| Title | 사용자는 선택작성을 골라 일기를 작성할 수 있다. |
| Detail | 로그인 후 메인 화면에 나타나는 달력에서 오늘 날짜를 선택하면 다이어리 기능이 실행된다. 일기 선택작성을 시작하면 사용자는 오늘의 기분이 어땠는지 묻는 질문에 ‘기쁨’, ‘우울’, ‘설렘’, ‘슬픔’ 등의 감정을 나타내는 단어를 고른다. 그러고 나서 오늘 무슨 일이 있었는지 묻는 질문에 ‘언제, 어디서, 누가(누구와), 무엇을, 어떻게, 왜’의 질문에 맞게 사용자가 보기에서 고르거나 원하는 대로 작성한다. 시스템은 사용자의 작성을 바탕으로 자동으로 산문 형식의 일기를 작성해주며 이 또한 사용자가 수정할 수 있다. |
| No. | USER\_RS\_002 |
| Title | 사용자는 직접작성을 골라 일기의 내용을 직접 작성할 수 있다. |
| Detail | 로그인 후 메인 화면에 나타나는 달력에서 오늘 날짜를 선택하면 다이어리 기능이 실행된다. 마찬가지로 오늘의 기분을 선택하고, 일기를 산문 형식으로 작성한다. |
| No. | USER\_RS\_003 |
| Title | 사용자는 내일 할 일을 메모할 수 있다. |
| Detail | 메인 화면에서 달력과 함께 내일 할 일을 보여준다. Todo List를 선택하면 내일 할 일을 정리하고 확인이 가능하다. 할 일을 이행하고 나서 사용자가 체크하면 해당 스케줄은 사라지고 남은 할 일들만 남게 된다. |
| No. | USER\_RS\_004 |
| Title | 사용자는 친구와 교환일기를 작성할 수 있다. |
| Detail | 교환일기장에 친구의 접근 권한을 부여하면 사용자와 함께 교환일기를 쓴다. 누가 먼저 쓸지, 일기 쓰는 기한 등은 두 사용자가 자유롭게 결정한다. |
| No. | USER\_RS\_005 |
| Title | 사용자는 자신의 습관과 일기 패턴을 분석한 시각화 자료를 볼 수 있다. |
| Detail | 시스템이 텍스트 분석을 통해 사용자의 일기 작성 패턴을 시각화해서 보여준다. 최근 한 달 간 많이 사용한 단어가 무엇인지 워드 클라우드 형태로 보여주고, 최근 한 달 동안 어떤 감정이 주를 이루었는지 보여준다. 시스템은 일기를 언제 마지막으로 작성했는지, 얼마나 꾸준하게 작성했는지, 어떤 요일에 많이 작성했는지를 데이터 처리해준다. |

# 4. System Requirements

## 4.1. Functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| No. | SYS\_RS\_001 | Related Requirements | USER\_RS\_001, USER\_RS\_005 |
| Title | 일기 선택사항을 데이터베이스에 저장하고 필요할 때 조회해 사용한다. | | |
| Description | 사용자가 선택한 기분, 사건에 관련된 내용을 데이터베이스에 저장한다. 그리고 일기 내용 조회, 수정, 삭제 또는 통계 자료 시각화를 위해서 접근이 필요할 때는 SQL Query문을 통해서 접근한다. | | |
| Necessity | 사용자가 작성한 내용을 영구적으로 보관, 수정, 삭제 또는 데이터 활용을 위해서는 반드시 데이터베이스에 저장하는 것이 필요하다. | | |
| No. | SYS\_RS\_002 | Related Requirements | USER\_RS\_002, USER\_RS\_005 |
| Title | 직접 작성한 일기를 데이터베이스에 저장하고 필요할 때 조회해 사용한다. | | |
| Description | 사용자가 작성한 기분, 사건에 관련된 내용을 데이터베이스에 저장한다. 그리고 일기 내용 조회, 수정, 삭제 또는 통계 자료 시각화를 위해서 접근이 필요할 때는 SQL Query문을 통해서 접근한다. | | |
| Necessity | 사용자가 작성한 내용을 영구적으로 보관, 수정, 삭제 또는 데이터 활용을 위해서는 반드시 데이터베이스에 저장하는 것이 필요하다. | | |
| No. | SYS\_RS\_003 | Related Requirements | USER\_RS\_003 |
| Title | 작성된 Todo List 항목들을 체크박스 형태로 제공한다. | | |
| Description | 사용자가 작성한 Todo List를 체크박스 형태로 제공하여 해야 할 일을 다 처리했을 경우 직접 체크해서 달성을 했다는 점을 시각적으로 쉽게 보여주게 한다. Html의 input 태그 checkbox를 활용해 기능을 구현한다. | | |
| Necessity | 해야 할 일을 달성한 사용자가 더욱 성취감을 높여 주기 위해서 정적인 텍스트가 뿐만 아니라 동적인 체크박스를 활용한다. | | |
| No. | SYS\_RS\_004 | Related Requirements | USER\_RS\_004 |
| Title | 교환일기를 작성하고자 하는 사용자에게 교환일기장을 생성해준다. | | |
| Description | 기존의 사용자에게 할당된 일기장을 제외하고 새로운 교환일기장을 생성한다. 그리고 생성된 교환일기는 사용자와 사용자가 지목한 다른 사용자, 이 두 명에게 접근 권한이 주어진다. | | |
| Necessity | 처음 사용자가 서비스를 사용하게 되면 기본적으로 일기가 할당되는데 이 일기는 오직 본인만 접근할 수 있는 권한을 가지게 된다. 그렇기 때문에 다른 사람이 접근할 수 있는 일기를 작성하려면 새로운 교환일기라는 Entity를 생성해야 한다. | | |
| No. | SYS\_RS\_005 | Related Requirements | USER\_RS\_004 |
| Title | 사용자는 유저 검색 및 초대기능을 활용한다. | | |
| Description | 교환일기 작성을 위해서 사용자가 자신의 파트너를 찾고 초대할 수 있는 기능을 구현한다. 검색은 아이디 혹은 닉네임과 같은 유저가 가지고 있는 unique key를 활용할 예정이고 검색 후 사용자에게 교환일기를 초대하는 기능을 구현한다. | | |
| Necessity | 교환일기를 작성하기 위해서 파트너를 구하기 위해서 반드시 필요한 기능이다. | | |
| No. | SYS\_RS\_006 | Related Requirements | USER\_RS\_005 |
| Title | 데이터베이스에 저장된 데이터를 활용해 시각화 한다. | | |
| Description | 사용자가 작성한 일기 본문에서 명사를 추출해 워드 클라우드 형태로 시각화 자료를 보여준다. 그리고 현재까지 일기를 작성한 횟수, 연속으로 일기를 작성한 횟수도 보여준다. 마지막으로 사용자의 최근 기분을 파이차트를 활용해 제시한다. | | |
| Necessity | 일기를 정적으로 쓰기만 하는 것이 아니라 사용자가 얼마나 꾸준히 일기를 쓰는지 시각적으로 보여줘 성취감을 고취시키고, 일기를 통해 어떤 단어를 주로 사용했는지와 어떤 기분을 주로 느꼈는지를 시각적으로 보여준다. | | |

## 4.2. Non-Functional Requirements

|  |  |
| --- | --- |
| No. | NF\_RS\_001 |
| Category | Product Requirement → Efficiency Requirement → Performance Requirement |
| Description | 제이콥 닐슨의 주장에 따라 사용자가 읽고 탐색하는 방식의 웹사이트는 반응시간이 1초이내를 유지해야 한다는 주장에 따라 **평균화면 반응시간을 1초이내로 설정**. 그리고 상대적으로 렌더링 시간이 긴 시각화 페이지의 경우 3초의 지연이 발생하면 사용자 25%가 웹 앱을 포기한다는 에버딘의 연구결과를 바탕으로 **일기분석 시각화 페이지 반응시간을 3초이내로 설정**.[[3]](#footnote-3) |
| No. | NF\_RS\_002 |
| Category | Product Requirement → Usability Requirement |
| Description | 회원가입을 한 사람은 자신의 일기에 대한 작성, 수정, 삭제를 자유롭게 할 수 있다. 또한 교환일기를 제외한 개인 일기는 접속한 본인만 사용 가능해야 한다. 교환일기 기능에 있어서는 교환일기 사용자 이외의 사용자는 접속할 수 없어야 한다. |
| No. | NF\_RS\_003 |
| Category | Product Requirement → Security Requirement |
| Description | **일기 전문은 데이터베이스에 인코딩을 통해 암호화 작업을 하고 저장한다**. 또한 페이지 렌더링에 사용되는 http 값 역시 **pk 값을 그대로 노출하지 않고 무작위의 코드를 사용**해 해커들이 추측성 request를 보내지 못하도록 한다. |
| No. | NF\_RS\_004 |
| Category | Organizational Requirement → Operational Requirement |
| Description | 서비스 운용에 사용자 트래픽이 증가해 성능이 감소할 때를 대비해 **쉽게 확장할 수 있도록 한다**. 데이터베이스와 가상서버를 따로 운용해 쉽게 서버를 옮길 수 있도록 한다. |
| No. | NF\_RS\_005 |
| Category | External Requirement → Ethical Requirement |
| Description | 시스템 관리자 또는 개발자는 사용자의 개인 정보와 일기 데이터를 절대 동의 없이 접근 및 사용하지 않는다. |

# 5. Requirements Specification

## 5.1. Usecase 1 - 선택 작성

|  |
| --- |
| SYS\_RS\_001 |
| **<그림 3. SYS\_RS\_001 간단한 개념도>** |
| 1. 사용자가 일기를 선택작성을 통해서 작성한다. 2. 작성된 일기를 SQL Query문을 통해서 데이터베이스에 접근하고 저장한다. 3. 요청에 맞는 SQL Query문을 통해서 데이터베이스에 접근하고 저장한다. 4. 일기의 변경된 사항 또는 시각화 자료를 사용자에게 출력한다. |

## 5.2. Usecase 2 - 직접 작성

|  |
| --- |
| SYS\_RS\_002 |
| **<그림 4. SYS\_RS\_002 간단한 개념도>** |
| 1. 사용자가 일기를 직접작성을 통해서 작성한다. 2. 작성된 일기를 SQL Query문을 통해서 데이터베이스에 접근하고 저장한다. 3. 요청에 맞는 SQL Query문을 통해서 데이터베이스에 접근하고 저장한다. 4. 일기의 변경된 사항 또는 시각화 자료를 사용자에게 출력한다. |

## 5.3. Usecase 3 - Todo List

|  |
| --- |
| SYS\_RS\_003 |
| **<그림 5. SYS\_RS\_002 간단한 개념도>** |
| 1. Html의 input태그를 사용자가 클릭 시 데이터베이스 값을 전송한다. 2. 데이터베이스의 값을 통해서 체크박스의 체크여부를 html을 통해 출력한다. |

## 5.4. Usecase 4 - 교환일기

|  |
| --- |
| SYS\_RS\_004 |
| **<그림 6. SYS\_RS\_003 간단한 개념도>** |
| 1. 사용자 A는 자산의 일기장에 접근할 수 있고 모든 권한을 가진다. 2. 사용자 A는 A와 B 공동소유의 교환일기에 모든 권한을 가진다. 3. 사용자 B는 A와 B 공동소유의 교환일기에 모든 권한을 가진다. 4. 사용자 B는 A의 일기장에 접근할 수 없고 어떠한 권한도 없다. |
| SYS\_RS\_005 |
| **<그림 7. SYS\_RS\_004 간단한 개념도>** |
| 1. 아이디, 닉네임과 같은 unique key를 통해 검색해 다른 사용자를 찾을 수 있다. 2. 검색한 결과를 통해서 사용자에게 교환일기 초대를 요청할 수 있다. |

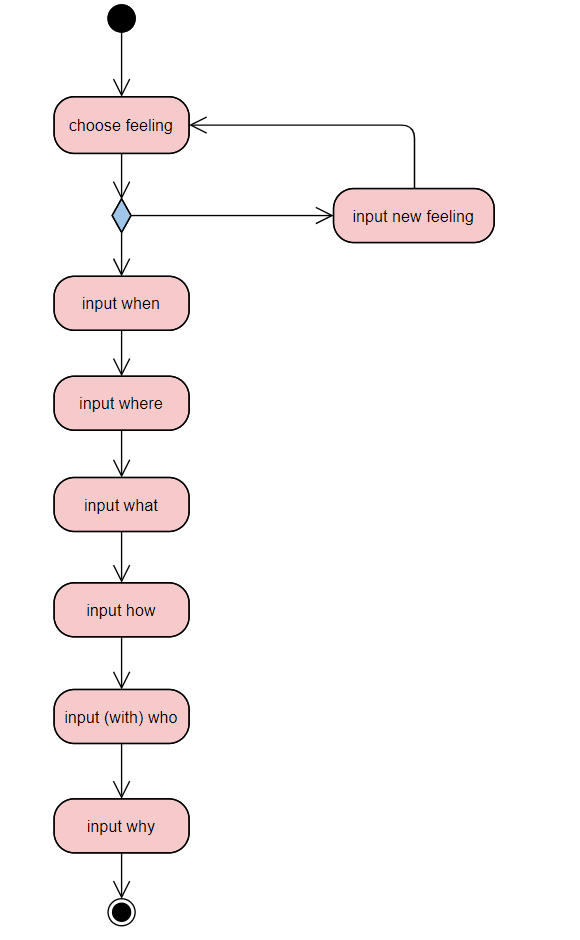
## 5.5. Usecase 5 - Analysis and Visualization

|  |
| --- |
| SYS\_RS\_006 |
| **<그림 8. SYS\_RS\_005 간단한 개념도>** |
| 1. 일기에 작성된 본문의 명사를 추출해 빈도수에 따라 워드 클라우드로 시각화 한다. 2. 지금까지 일기를 작성한 횟수, 연속으로 작성한 횟수를 계산해 제공한다. 3. 최근 기분을 파이차트 형태로 사용자가 쉽게 확인할 수 있게 출력한다. |

# 6. Requirements Analysis

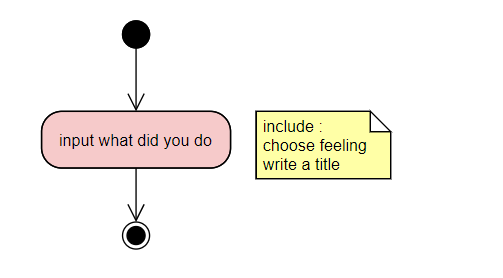
## 6.1. Behavioral Analysis

### 6.1.1. Usecase 1 - 선택 작성



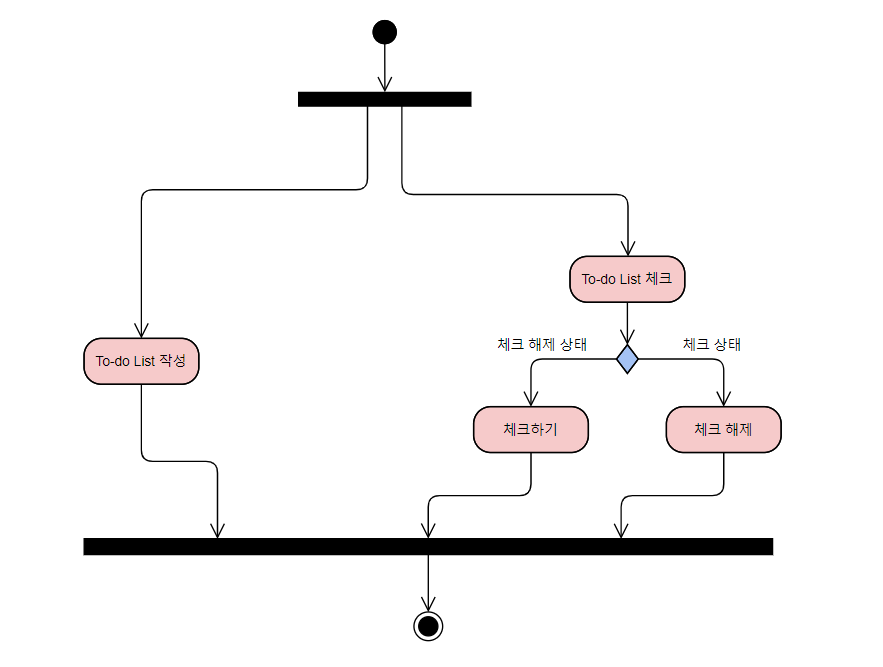
**<그림 9. 선택 작성 Activity Diagram>**

### 6.1.2. Usecase 2 - 직접 작성



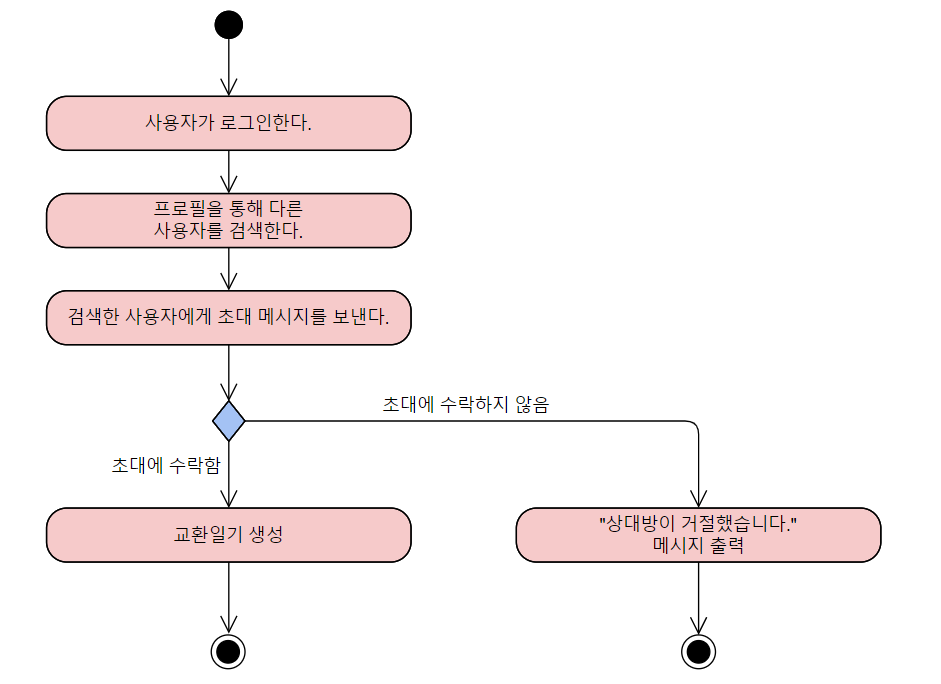
**<그림 10. 직접 작성 Activity Diagram>**

### 6.1.3. Usecase 3 - Todo List



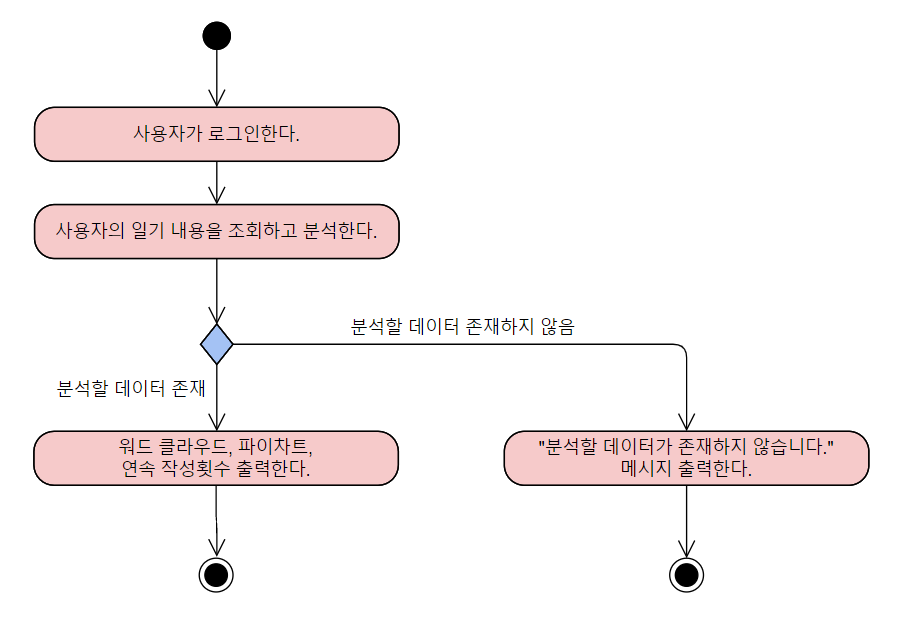
**<그림 11. Todo List Activity Diagram>**

### 6.1.4. Usecase 4 - 교환 일기



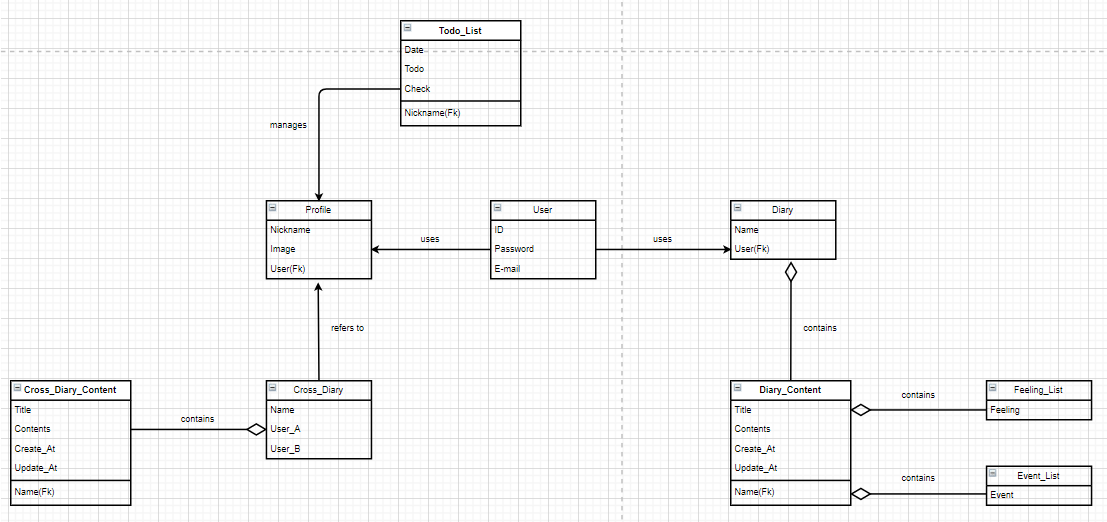
**<그림 12. 교환 일기 Activity Diagram>**

### 6.1.5. Usecase 5 - Analysis and Visualization



**<그림 13. Analysis and Visualization Activity Diagram>**

## 6.2. Structural Analysis



<그림 14. Domain Model Diagram>

### 6.2.1 Briefly describe each analysis class

|  |  |
| --- | --- |
| Class Name | User |
| Description | “하루 끝”을 사용하는 사용자 |
| Relation | Association to Profile, Diary |
| Attribute | ID, Password, E-mail |

|  |  |
| --- | --- |
| Class Name | Profile |
| Description | 사용자의 개인정보 저장 |
| Relation | Association to User, Todo\_List, Cross\_Diary |
| Attribute | Nickname, Image, User(FK) |

|  |  |
| --- | --- |
| Class Name | Diary |
| Description | 사용자의 개인 다이어리를 관리 |
| Relation | Association to User, Diary\_Content |
| Attribute | Name, User(FK) |

|  |  |
| --- | --- |
| Class Name | Diary\_Content |
| Description | 사용자의 개인 다이어리의 세부 내용 |
| Relation | Association to Diary, Feeling\_List, Event\_List |
| Attribute | Title, Contents, Create\_At, Update\_At |

|  |  |
| --- | --- |
| Class Name | Feeling\_List |
| Description | 선택작성에 필요한 Feeling관련 선택목록 |
| Relation | Association to Diary\_Content |
| Attribute | Feeling |

|  |  |
| --- | --- |
| Class Name | Event\_List |
| Description | 선택작성에 필요한 Event관련 선택목록 |
| Relation | Association to Diary\_Content |
| Attribute | Event |

|  |  |
| --- | --- |
| Class Name | Todo\_List |
| Description | 사용자의 Todo\_List를 관리 |
| Relation | Association to Profile |
| Attribute | Date, Todo, Check, Nickname(FK) |

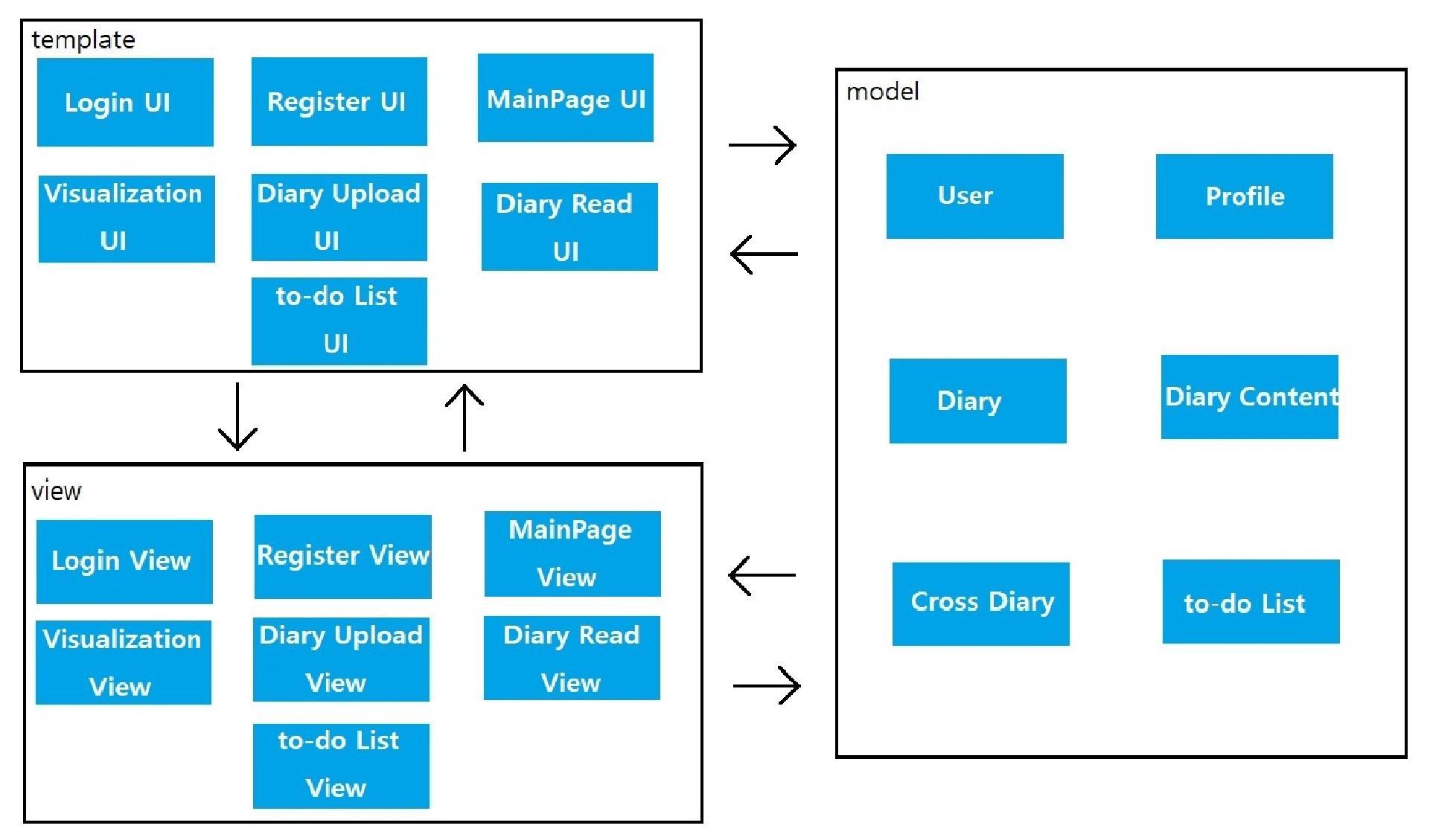
|  |  |
| --- | --- |
| Class Name | Cross\_Diary |
| Description | 교환일기를 공유하는 2명의 사용자 및 교환일기 관리 |
| Relation | Association to Profile, Cross\_Diary\_Content |
| Attribute | Name, User\_A, User\_B |

|  |  |
| --- | --- |
| Class Name | Cross\_Diary\_Content |
| Description | 교환일기의 세부내용 |
| Relation | Association to Cross\_Diary |
| Attribute | Title, Contents, Create\_At, Update\_At, Name(FK) |

## 

# 7. Architectural Design

## 7.1. Logical View / Architecture



**<그림 15. Client-Server Architectural Design>**

### 7.1.1. Layer Description

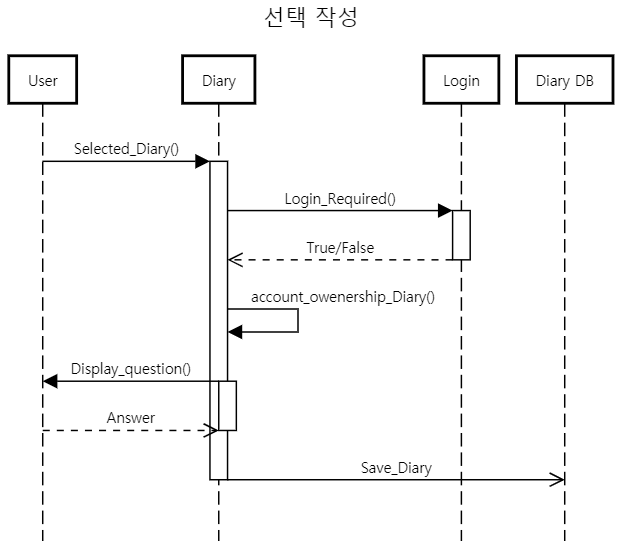
Architectural Design은 client-server model을 통해 디자인하였다. 웹 개발에 사용될 django에서는 MVT 형식을 사용해 개발을 진행하게 되는데 이는 Java spring에서 사용하는 MVC 패턴과 비슷하다. Model은 데이터베이스와 상호작용하는 인터페이스 역할이고 View는 로직을 처리하는 역할로 spring의 controller와 같은 역할이다. Template은 html과 같은 사용자에게 출력되는 화면을 나타내는 것으로 spring의 view와 같은 역할이다.

Django MVT 구조에서 View와 template은 보통 1대1로 대응되고 model은 사용자 내용을 처리하는 user, profile로 구성하고 일기 내용을 처리하는 diary와 diary\_content로 구성했다. 또한 교환일기는 cross diary로 설정했고 Todo List의 기능은 to-do list 테이블을 작성해 관리한다.

# 8. Design Modeling

## 8.1. Usecase 1 - 선택 작성

### 8.1.1. Design Sequence Diagram

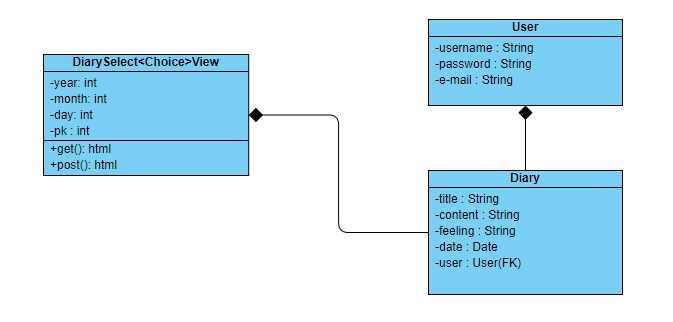


<그림 16. 선택 작성 Sequence Diagram>

#### 8.1.1.1. <scenario 1> 선택 작성

1. 사용자는 로그인을 하고, 사용자만의 시스템을 제공받는다.
2. 사용자가 오늘 날짜를 달력에서 선택한다.
3. 시스템은 일기를 선택 작성으로 쓰고 싶은지 직접 작성으로 쓰고 싶은지 묻는다. 사용자는 선택 작성을 클릭한다.
4. 사용자는 오늘의 기분을 보기에서 고른다. 보기에 없는 다른 감정을 넣고 싶다면 기타에서 작성하면 된다.
5. 육하원칙에 맞게 질문이 하나씩 나온다. 사용자는 보기에서 선택하거나 기타에서 작성한다.
6. 시스템은 사용자의 답변을 일기 형태로 저장한다.

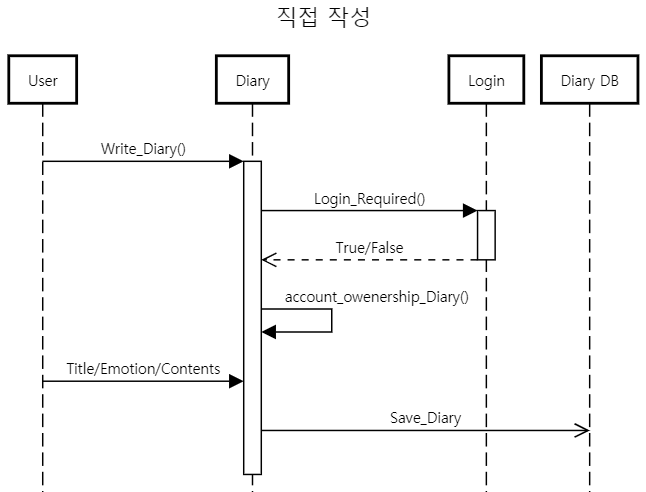
### 8.1.2. Design Class Diagram



<그림 17. 선택 작성 Class Diagram>

## 8.2. Usecase 2 - 직접 작성

### 8.2.1. Design Sequence Diagram

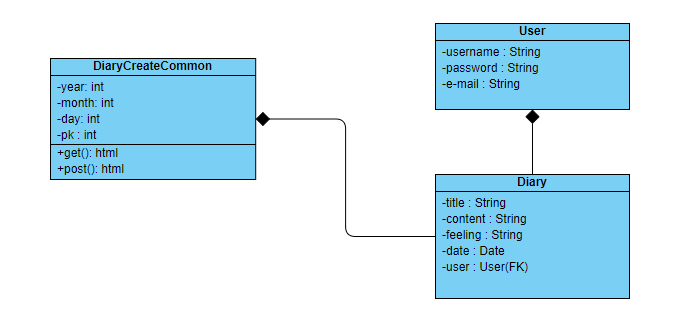


**<그림 18. 직접 작성 Sequence Diagram>**

#### 8.2.1.1. <scenario 2> 직접 작성

1. 사용자는 로그인을 하고, 사용자만의 시스템을 제공받는다.
2. 사용자가 오늘 날짜를 달력에서 선택한다.
3. 시스템은 일기를 선택 작성으로 쓰고 싶은지 직접 작성으로 쓰고 싶은지 묻는다. 사용자는 직접 작성을 클릭한다.
4. 사용자는 제목을 작성하고, 오늘의 기분을 카테고리에서 고르고, 마음껏 일기를 작성한다.
5. 시스템은 사용자의 일기를 저장한다.

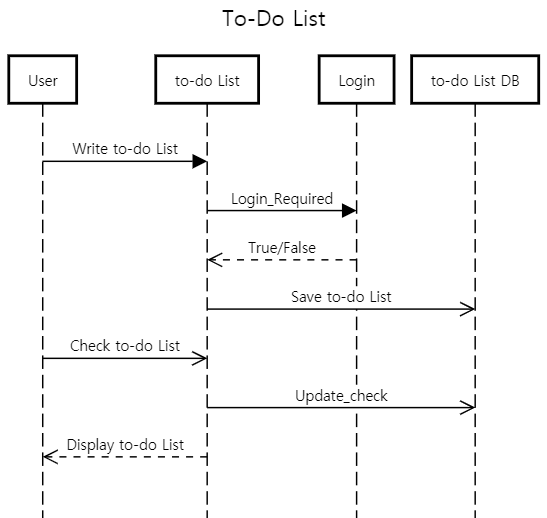
### 8.2.2. Design Class Diagram



<그림 19. 직접 작성 Class Diagram>

## 8.3. Usecase 3 - Todo List

### 8.3.1. Design Sequence Diagram



<그림 20. Todo List Sequence Diagram>

#### 8.3.1.1. <scenario 3-1> Todo List 추가

1. user가 Todo List 빈칸에 할일을 적는다.
2. user가 Todo List 할일을 적은 뒤 enter를 누른다.

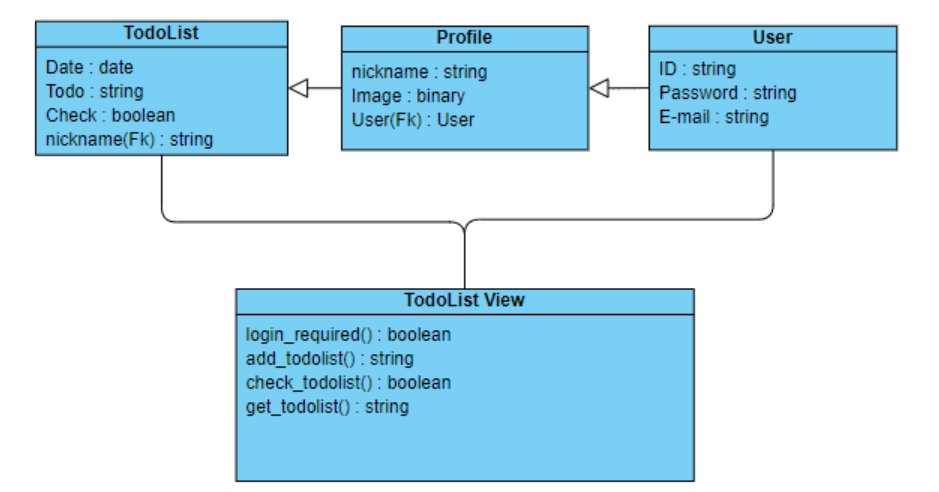
#### 8.3.1.2. <scenario 3-2> Todo List 삭제

user가 Todo List 체크표에 체크한다.

#### 8.3.1.3. <scenario 3-2> Todo List 수정

1. user가 Todo List 수정 버튼을 누른다.
2. user가 Todo List 창에 수정할 내용을 입력한다.
3. user가 수정 완료 버튼을 누른다.

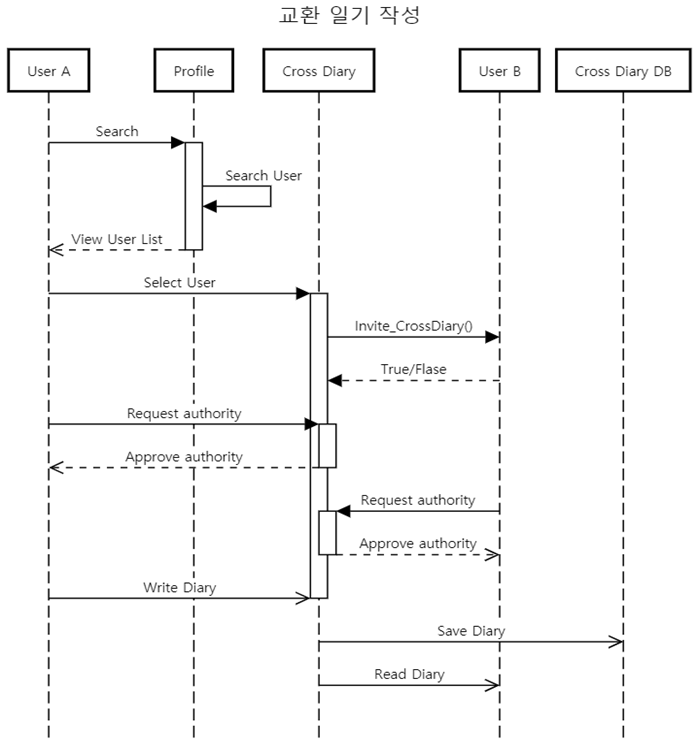
### 8.3.2. Design Class Diagram



**<그림 21. Todo List Class Diagram>**

## 8.4. Usecase 4 - 교환 일기

### 8.4.1. Design Sequence Diagram

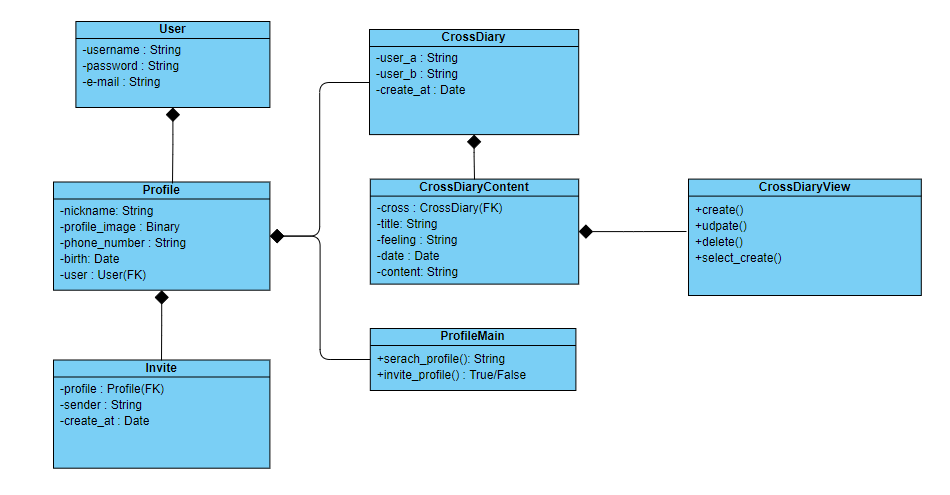


<그림 22. 교환일기 Sequence Diagram>

#### 8.4.1.1 <scenario 4> 교환 일기

1. 사용자가 로그인을 하고, 사용자만의 시스템을 제공받는다.
2. (사용자가 교환일기를 함께 쓰고 싶은 다른 사용자를 검색한다. 그리고 초대 메시지를 보낸다.)
3. (초대를 수락했다는 전제 하에)제공받은 교환 일기장에서 상대방의 일기를 확인하거나 사용자가 새로운 일기를 쓴다.

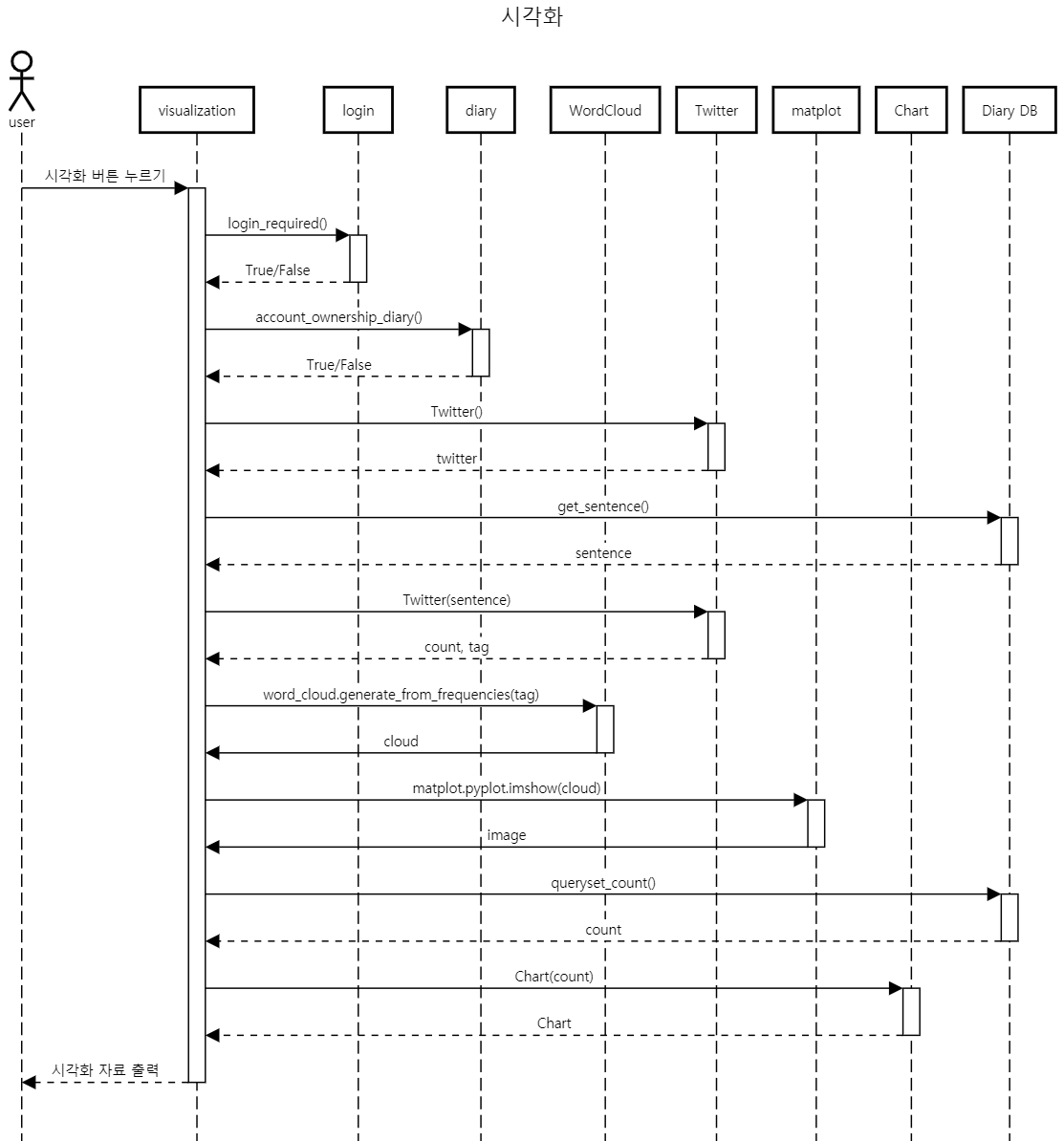
### 8.4.2. Design Class Diagram



<그림 23. 교환 일기 Class Diagram>

## 8.5. Usecase 5 - Analysis and Visualization

### 8.5.1. Design Sequence Diagram



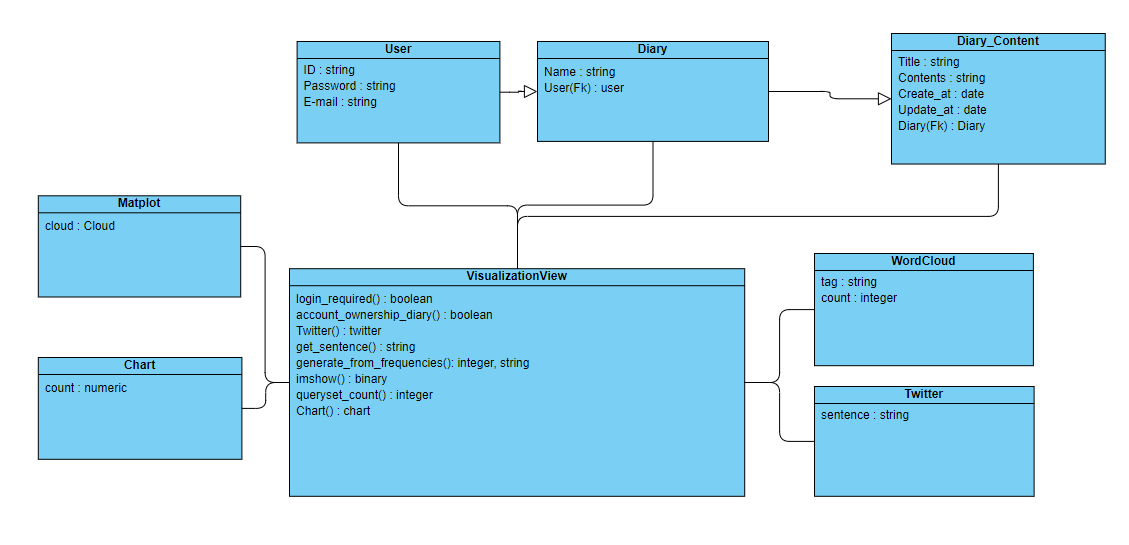
**<그림 24. 시각화 Sequence Diagram>**

#### 8.5.1.1 <scenario 5> 사용자 기록 시각화

1. user가 시각화버튼을 클릭한다.
2. 시각화 된 이미지를 보여준다.
3. 홈버튼을 누르면 홈으로 되돌아간다.

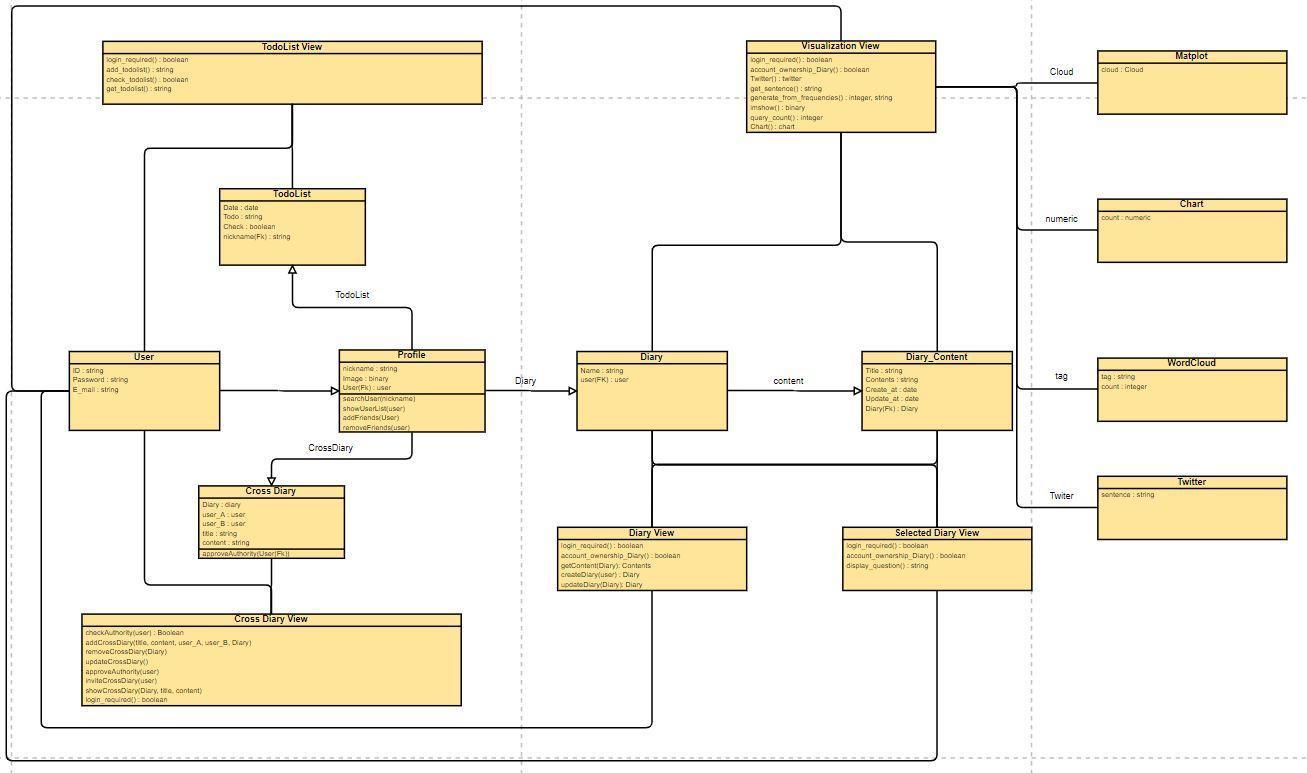
.

### 8.5.2 Design Class Diagram



<그림 25. 시각화 Class Diagram>

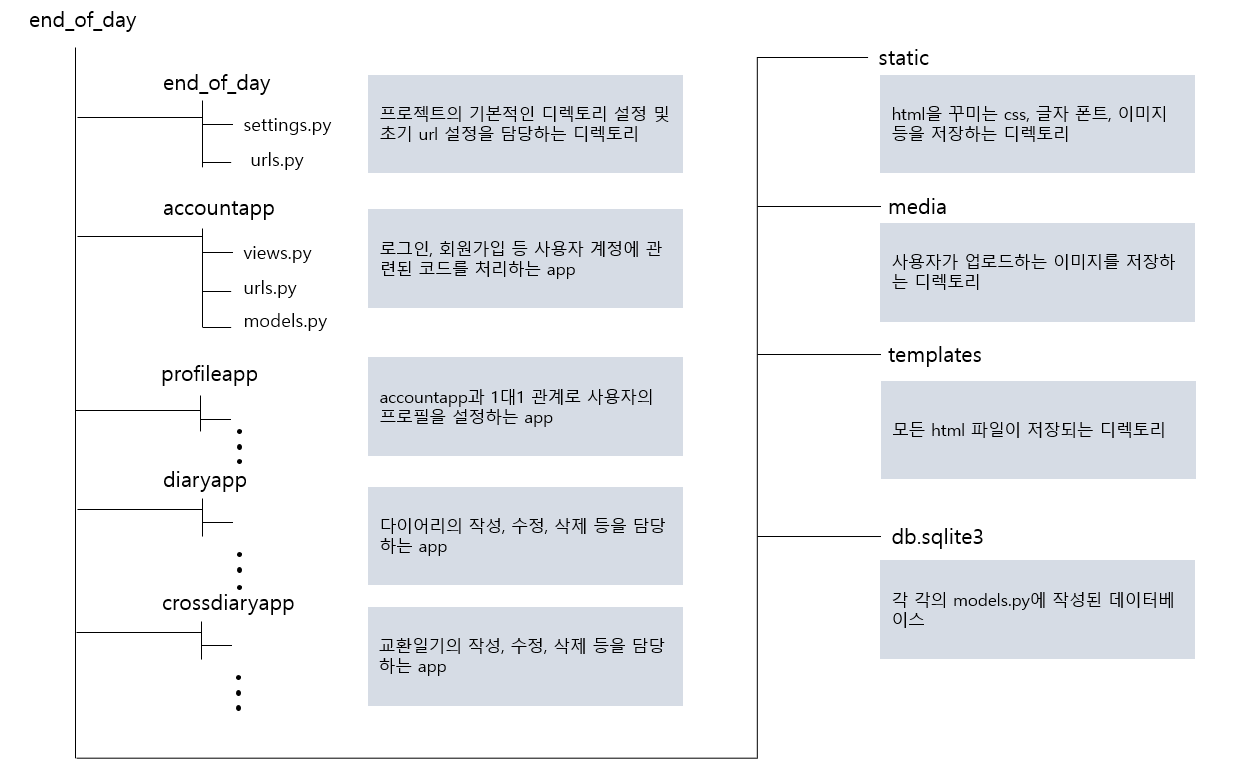
## 8.6. Combined Design Class Diagram



<그림 26. Design Class Diagram>

# 9. Implementation

## 9.1. Implementation Specific



<그림 27. 프로젝트 구조>

위 그림은 프로젝트의 전체적인 구조로 크게 각 기능을 담당하는 app과 static, media와 같은 프론트 쪽의 디렉토리 그리고 데이터베이스로 구성되어 있다.

### 9.1.1. end\_of\_day

프로젝트의 전반적인 변수와 초기 URL을 설정하는 디렉토리다.

#### 9.1.1.1. settings.py

프로젝트에 사용되는 변수를 지정해주는 파일로 static, media 경로 그리고 base\_dir 등을 설정해준다. 뿐만 아니라 미들웨어, 데이터베이스의 설정 또한 하며 배포 시 호스팅 되는 도메인 주소 등도 변경이 가능하다.

#### 9.1.1.2. urls.py

웹에서 가장 기초가 되는 URL을 설정하는 파일로 도메인주소/<url>을 설정하는 것부터 시작한다. 기본적으로 관리자 페이지를 제공하고 있기 때문에 도메인주소/admin을 제공한다.

### 9.1.2. accountapp

#### 9.1.2.1. views.py

흔히 사용하는 MVC 구조에서 Controller가 하는 역할을 django에서는 view가 맡아서 하는데 url과 대응되어 사용자들에게 request 받아 적절한 template을 사용자에게 출력해주는 역할을 한다. Accountapp의 view는 로그인, 회원가입 등 사용자의 계정에 관련된 get, post를 처리한다.

#### 9.1.2.2. urls.py

View와 대응하여 사용자가 웹 서버에 get request를 요청했을 때, 해당 url과 view를 연결해주는 파일이다. 기본적인 로그인, 로그아웃, 회원가입 등으로 이루어져 있다.

#### 9.1.2.3. models.py

데이터베이스 구조를 ORM을 통해 쉽게 정의할 수 있는 파일로 accountapp에서는 사용자 class인 user를 정의하는데 이는 기본적으로 내장되어 있는 User를 활용하였다.

#### 9.1.2.4. forms.py

회원가입, 로그인과 같이 사용자의 입력을 받아야 할 때 사용자에게 적절한 폼을 출력해주고 사용자가 입력한 값을 model에 전달하는 역할을 한다.

### 9.1.3. profileapp

#### 9.1.3.1. views.py

마찬가지로 controller 역할의 view로 profileapp에서는 사용자가 profile을 작성하고 수정, 삭제할 수 있는 기능을 제공하고 교환일기 작성 시 사용자를 초대할 수 있도록 검색 기능을 제공하는 역할이다. 또한 다른 사용자의 profile nickname을 통해 검색하는 기능도 제공한다.

#### 9.1.3.2. urls.py

위의 view와 대응되는 URL로 프로필 작성, 프로필 확인, 수정, 검색과 같은 url이 설정되어 있다.

#### 9.1.3.3. models.py

사용자 프로필에 관련된 ORM이 설정되어 있는데 profile 테이블은 먼저 사용자 계정인 User를 1대1로 대응되는 관계이며 닉네임, 성별, 생일 등의 정보가 저장된다.

#### 9.1.3.4. forms.py

위에 정의된 profile 테이블에 사용자가 프로필 생성, 수정할 때 사용하는 폼을 작성해주는 역할의 파일이다.

### 9.1.4. diaryapp

#### 9.1.4.1. views.py

다이어리의 직접작성, 선택작성과 같은 다이어리 관련 기능을 구현하고 있는 view파일이다. 뿐만 아니라 다이어리 수정, 삭제 등 다이어리 관련된 거의 대부분의 기능을 포함하고 있다. 그리고 다이어리 작성 상태를 달력형태로 확인할 수 있도록 구현하고 있다.

#### 9.1.4.2. urls.py

위의 다이어리 관련 view와 대응되는 URL로 위에서 작성된 view에 맞게 URL이 mapping되어 있다. 그리고 다이어리의 소유를 확인하기 위해서 다이어리 pk값 등을 URL의 파라미터 값으로 설정하고 있다.

#### 9.1.4.3. models.py

Diary라는 테이블이 작성되어 있는데 작성자의 정보가 user로 User 테이블의 객체가 저장되고 제목, 기분, 날짜, 내용으로 구성되어 있다.

#### 9.1.4.4. forms.py

사용자가 Diary를 직접작성하거나 수정할 때 사용되는 폼을 만들고 출력해주는 파일이다.

### 9.1.5. crossdiaryapp

#### 9.1.5.1. views.py

사용자가 CrossDiary를 직접작성하거나 수정할 때 사용되는 폼을 만들고 출력해주는 파일이다. 또한 교환일기를 소유하고 있는 소유자만 교환일기에 접근할 수 있도록 한다.

#### 9.1.5.2. urls.py

Crossdiary에 연결된 url이 저장되는 파일로 소유자만 접근할 수 있도록 사용자의 nickname과 비교하는 nickname을 url에 설정한다.

#### 9.1.5.3. models.py

CrossDiary를 통해 교환일기 사용자의 정보와 생성 정보를 담는 모델을 생성한다. 그리고 일기 내용은 CrossDiary를 외래키로 받아 CrossDiaryContent에 저장된다.

### 9.1.6. static

Html를 꾸미기 위한 base.css 파일, 그리고 여러가지 이미지 파일 등 전체적인 UI를 예쁘게 하기 위한 요소들이 저장되어 있다.

### 9.1.7. media

사용자가 프로필을 작성할 때 자신의 사진을 올리는 경우 사진은 이 디렉토리에 저장되게 된다. 그리고 다시 사용자가 사진을 확인하는 request를 했을 때 media 디렉토리로 사진을 출력해준다.

### 9.1.8. templates

위에 app에서 작성된 URL, view와 대응되는 모든 html 파일이 저장되는 디렉토리다. 하위 디렉토리로 각각의 app 이름으로 작성하여 쉽게 분류할 수 있도록 구성하였다.

### 9.1.9 db.sqlite3

각 app에서 models.py로 만들어진 테이블의 구조로 데이터베이스가 구성되어 있으며 select, insert, update, delete와 같은 DML이 일어났을 때 실제 데이터가 변화한다.

## 9.2. Message Format (or any other contents)

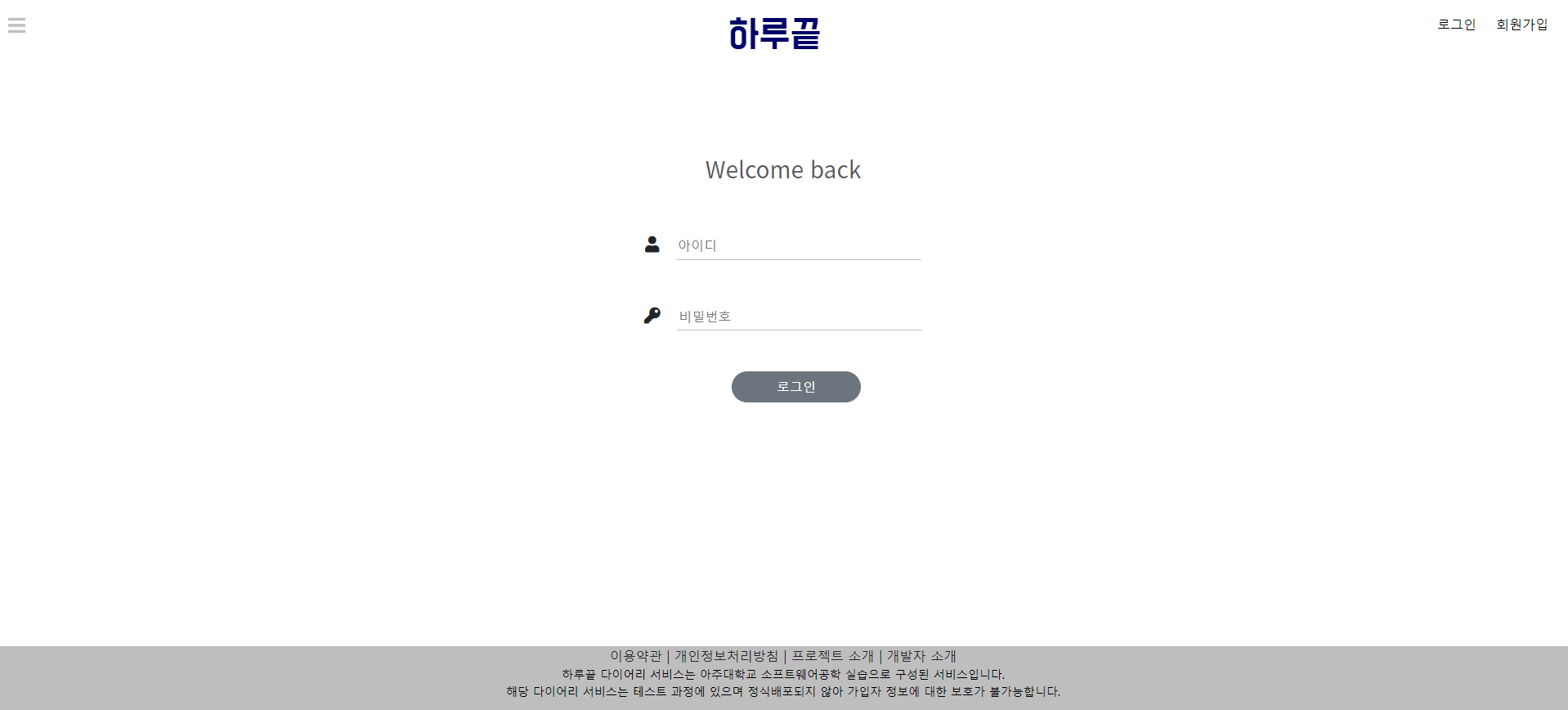
django에서 기본적으로 제공해주는 message format을 활용한다. 메시지의 종류로는 info, success, warning, error의 메시지 종류가 있으며 httpRequest 인스턴스를 통해 남기기 때문에 1회성이다.

## 9.3. System/Data Dependencies & Requirements

데이터 의존성으로는 먼저 User 테이블과 Profile, Diary 테이블이 있다. User 테이블과 Profile 테이블의 경우 Profile 테이블이 User 테이블에 의존하고 있으며 User 의 데이터가 삭제되었을 때 해당 Profile 데이터 역시 삭제되는 CASCADE 관계다. Diary 테이블은 User 테이블에 1대 다 관계로 profile 테이블과 똑같이 User테이블의 정보가 삭제되었을 때 Diary 테이블의 데이터도 같이 삭제되는 CASCADE 관계다.

# 10. UI Design

## 10.1. account

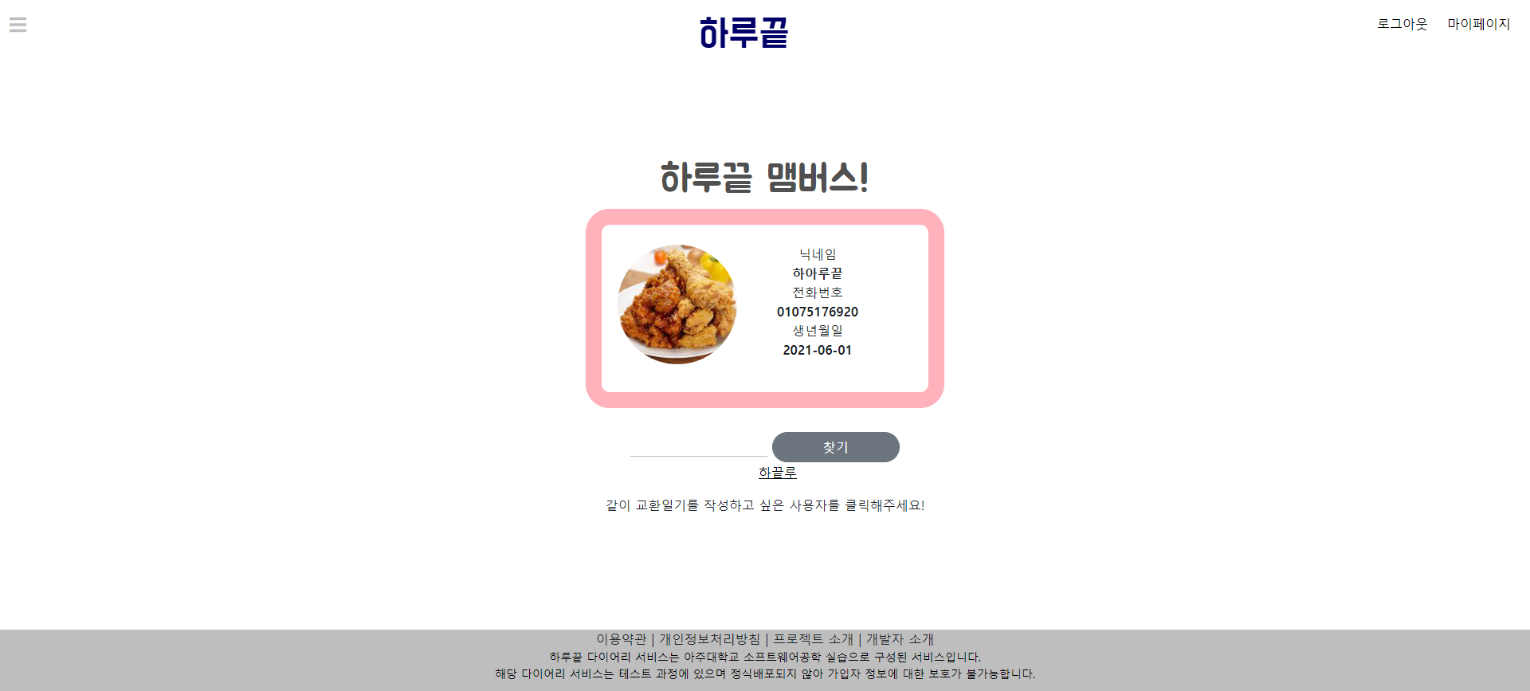


<그림28. 로그인 페이지>

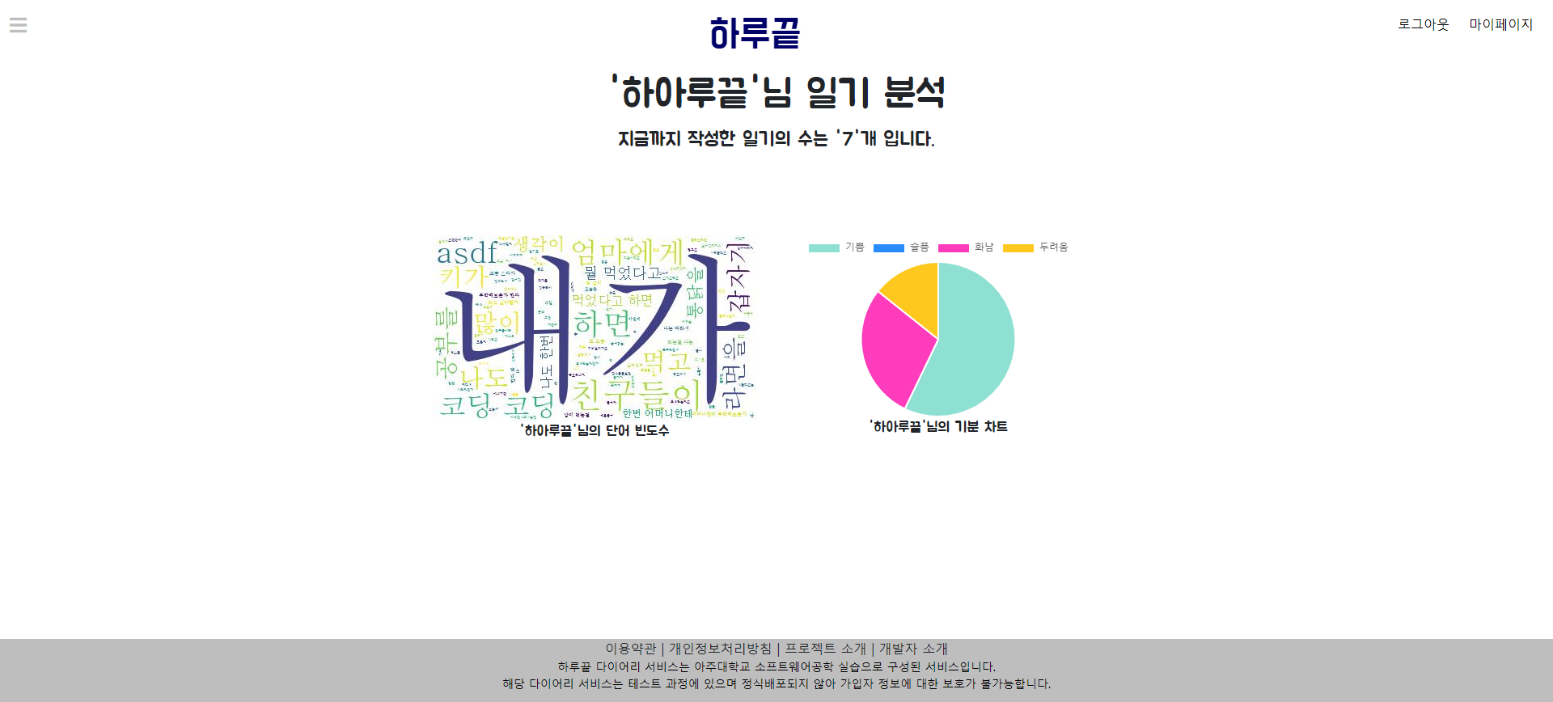


<그림 29. 회원가입 페이지>

## 10.2. profile



<그림 30. 프로필 메인 화면>

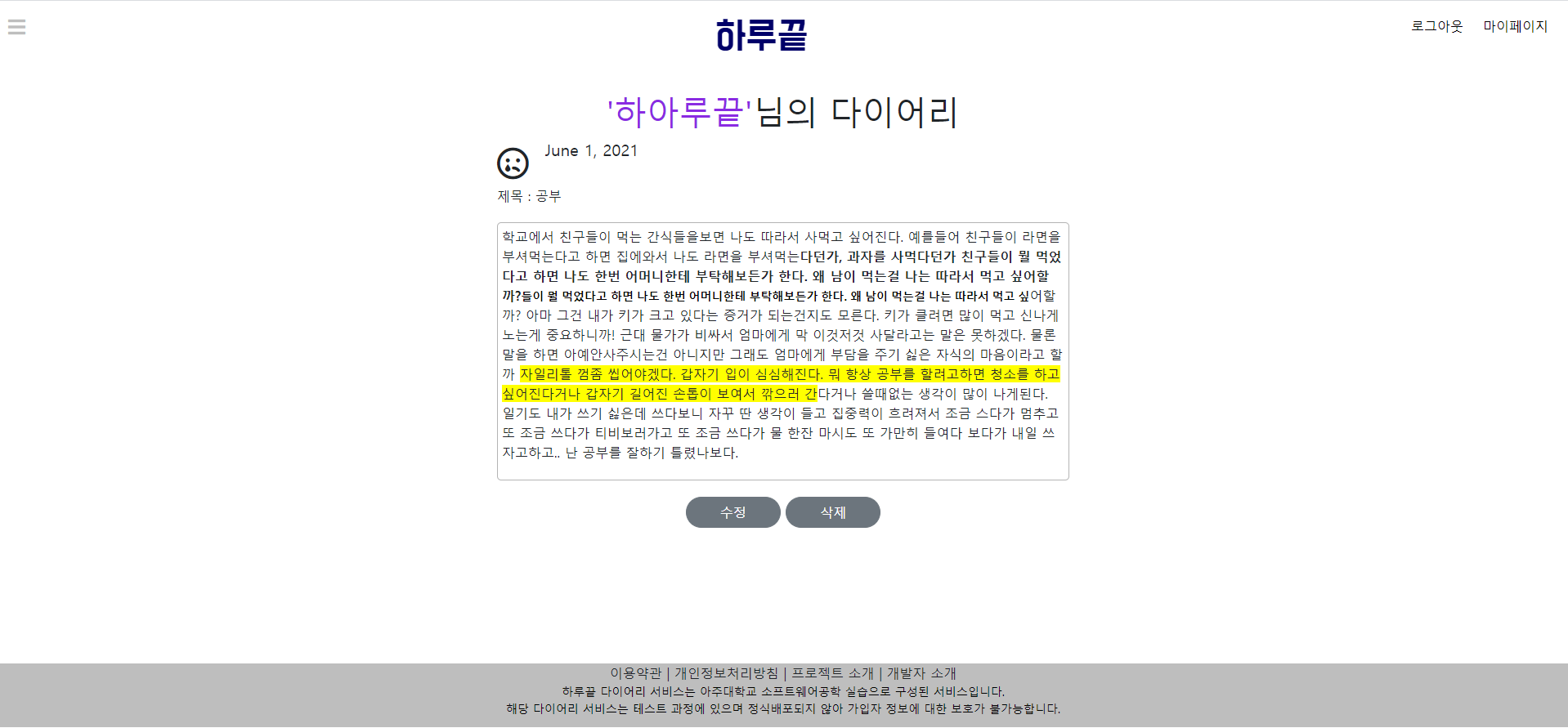


<그림 31. 일기분석 페이지>

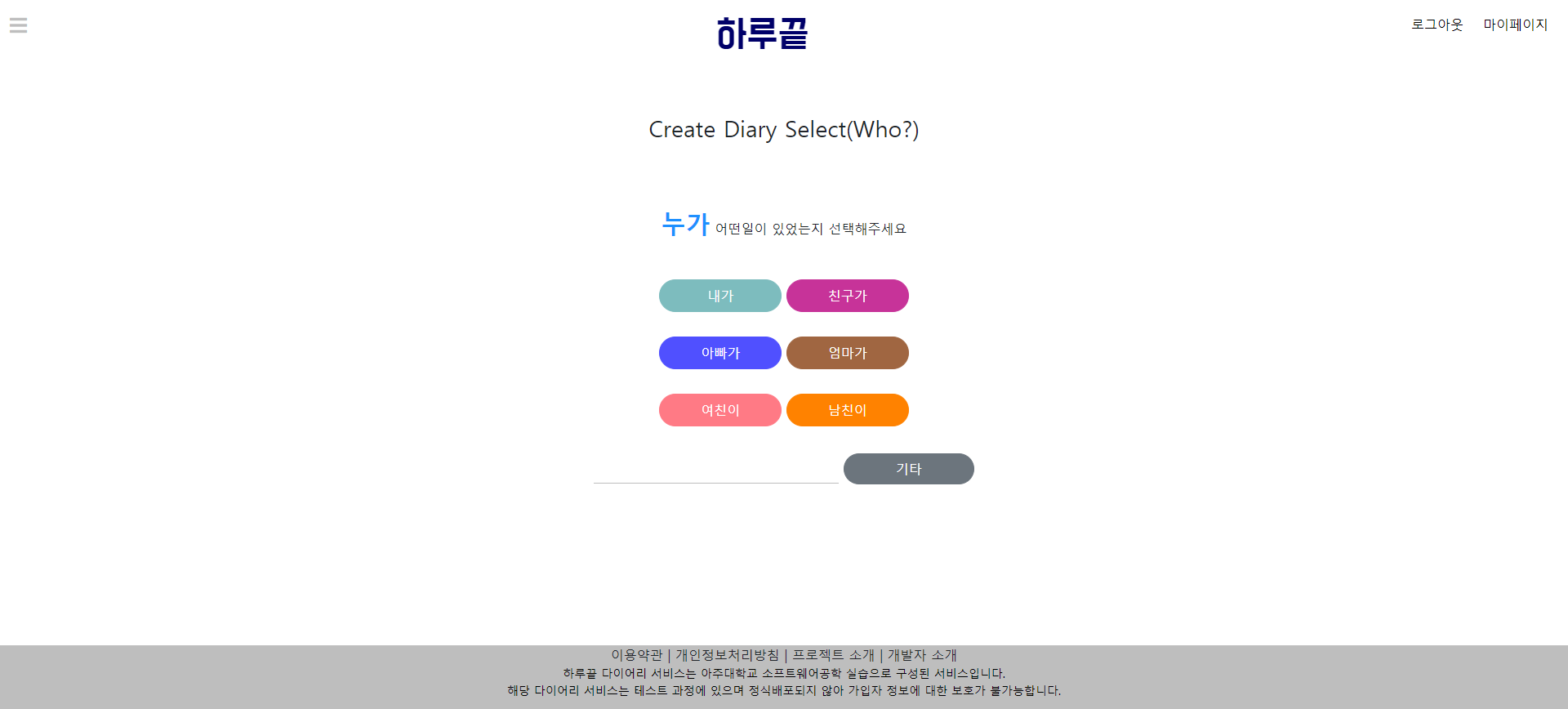
## 10.3. diary



<그림 32. 일기 메인 화면>



<그림 33. 일기읽기 페이지>



<그림 34. 일기 선택작성 페이지>

## 10.4. cross diary



<그림 35. 교환일기 리스트 페이지>

# 11. Test Plan & Result

이번 프로젝트에서 다이어리 웹 서비스를 출시하는 만큼 사용자가 오류 없이 잘 사용할 수 있는 서비스를 구현하기 위해 test를 진행한다. 각 기능별로 크고 작은 버그가 발생하는지 확인하기 위해 다양한 방법으로 request를 하고 response를 관찰할 것이다.

## 11.1. Validation criteria & Scope

### 11.1.1 다이어리 선택작성

다이어리 선택작성은 직접작성과 다르게 사용자가 크게 입력하는 값이 적기 때문에 오류발생이 적을 것으로 예상된다. 하지만 선택해야하는 문항 수가 많은 만큼 뒤로 가기 또는 작성 취소 등과 같이 임의로 작성을 그만두는 상황에 대해서 확인할 예정이다.

### 11.1.2. 다이어리 직접작성

다이어리 직접작성에서 사용자가 내용을 작성할 때 적절한 폼이 작성되고 잘 데이터베이스에 저장되는지를 확인할 예정이다. 그리고 최대한 사용자가 오류를 범할 수 있는 입력 값은 시스템에서 자동으로 처리하도록 구현할 예정이다.

### 11.1.3. to-do list

Todo list는 이번 구현 범위에 포함하지 않는다.

### 11.1.4. 교환일기

교환일기의 가장 주의해야할 부분은 바로 일기 작성 권한 문제다. 하나의 일기를 한 사람이 소유하는 것이 아니라 두 사람이 소유하고 관리하기 때문에 발생하는 문제를 위주로 테스트할 예정이다.

### 11.1.5. 시각화

시각화는 사용자의 일기를 전부 조회하고 분석해야 하는 것이기 때문에 데이터베이스에서 해당 내용을 조회하고 전처리를 하는 과정에서 버그가 발생하는지 여부를 집중적으로 테스트 해야 한다.

## 11.2. Test Plan

### 11.2.1. Software to be Tested

Django 환경에서 페이지가 정상적으로 렌더링 되는지 여부와 데이터베이스와의 데이터 이동이 유기적으로 되는지 여부를 테스트한다. 또한 사용하는 모듈이 Django 환경에서 잘 작동하는지 또한 테스트한다.

### 11.2.2. Testing Strategy

#### 11.2.2.1. Unit testing

- accountapp: 사용자의 계정을 담당하는 부분으로 로그인, 회원가입과 같이 사용자의 정보가 잘 저장이 되는지 여부를 중점적으로 보고 username을 통한 페이지 렌더링에 URL 값에 request가 잘되는지 여부를 확인한다.

- Profileapp: 사용자가 직접 프로필을 설정함으로써 사용되는 프로필로 검색, 시각화 등의 기능이 잘 작동하는지 여부를 중점적으로 확인한다.

- Diaryapp: 다이어리 직접작성과 선택작성이 잘 작동하는지 여부와 함께 다이어리를 소유한 사람만 자신의 다이어리를 접근할 수 있는지 여부를 확인한다.

- CrossDiaryapp: 교환일기 생성과 삭제, 수정에 대한 변화가 잘 작동하고 데이터베이스에 반영되는지 여부와 두 사람에게 정확하게 권한이 부여되는지 여부를 확인한다.

#### 11.2.2.2. Integration / System testing

전반적인 기능들이 구현되어 있는 app들과 사용자에게 출력되는 templates이 잘 꾸며지고 깨지는 화면은 없는지 그리고 출력되지 않는 이미지는 없는지 등을 확인하고 모든 app에서 데이터베이스와 잘 통신하는 지 등을 확인해 완성도 높은 서비스를 구현한다.

### 11.2.3. Testing Tools and Environment

따로 testing tool은 사용하지 않고 Django 자체 디버깅을 활용해서 문제가 발생하는 코드 라인과 오류문을 통해서 test 후 발생하는 코드를 수정할 예정이다.

## 11.3. Integration/System Tests

### 11.3.1. 다이어리 선택작성

|  |  |  |
| --- | --- | --- |
| TEST INFORMATION | | |
| Test Case Number : T\_01 | | |
| Functionality Tested : 다이어리 선택작성 | | |
| Test Type : Positive | | |
| Precondition : 사용자가 로그인 상태 | | |
| TEST STEPS | | |
| # | STEP | Expected Result |
| 1 | 사용자가 일기작성 페이지에 진입한다. | 선택작성과 직접작성 버튼 두개를 출력한다. |
| 2 | 사용자가 선택작성을 누른다 | 기분을 선택할 수 있는 화면을 출력해준다. |
| 3 | 기분을 사용자가 선택하거나 값을 입력한다. | ‘누가’에 대한 질문 화면을 출력해준다. |
| 4 | ‘누구’에 대한 내용을 선택하거나 값을 입력한다. | ‘언제’에 대한 질문 화면을 출력한다. |
| 5 | ‘언제’에 대한 내용을 선택하거나 값을 입력한다. | ‘어디서’에 대한 질문 화면을 출력한다. |
| 6 | ‘어디서’에 대한 내용을 선택하거나 값을 입력한다. | ‘무엇을’에 대한 질문 화면을 출력한다. |
| 7 | ‘무엇을’에 대한 내용을 작성한다. | ‘어떻게’에 대한 질문 화면을 출력한다. |
| 8 | ‘어떻게’에 대한 내용을 작성한다. | ‘왜’에 대한 질문 화면을 출력한다. |
| 9 | ‘왜’에 대한 내용을 작성한다. | 계속 작성 여부에 대한 화면을 출력한다. |
| 10 | 아니요 버튼을 누른다. | 작성된 일기를 보여주는 화면으로 이동한다. |

### 11.4.2. 다이어리 직접작성

|  |  |  |
| --- | --- | --- |
| TEST INFORMATION | | |
| Test Case Number : T\_02 | | |
| Functionality Tested : 다이어리 직접작성 | | |
| Test Type : Positive | | |
| Precondition : 사용자가 로그인 상태 | | |
| TEST STEPS | | |
| # | STEP | Expected Result |
| 1 | 사용자가 일기작성 페이지에 진입한다. | 선택작성과 직접작성 버튼 두개를 출력한다. |
| 2 | 사용자가 직접작성을 누른다 | 다이어리를 작성하는 폼을 출력한다. |
| 3 | 사용자는 폼에 맞게 제목, 기분, 내용을 작성하고 생성 버튼을 누른다. | 작성된 일기를 보여주는 화면으로 이동한다. |
| TEST INFORMATION | | |
| Test Case Number : T\_03 | | |
| Functionality Tested : 다이어리 직접작성 | | |
| Test Type : Negative | | |
| Precondition : 사용자가 로그인 상태 | | |
| TEST STEPS | | |
| # | STEP | Expected Result |
| 1 | 사용자가 일기작성 페이지에 진입한다. | 선택작성과 직접작성 버튼 두개를 출력한다. |
| 2 | 사용자가 직접작성을 누른다 | 다이어리를 작성하는 폼을 출력한다. |
| 3 | 사용자는 폼에 맞게 제목, 기분, 내용을 작성하지 않고 생성 버튼을 누른다. | 작성되지 않은 폼을 작성하라는 메시지를 출력한다. |

### 11.4.3. todo list

테스트 범위 아님

### 11.4.4. 교환일기

|  |  |  |
| --- | --- | --- |
| TEST INFORMATION | | |
| Test Case Number : T\_04 | | |
| Functionality Tested : 교환일기 작성 초대 | | |
| Test Type : Positive | | |
| Precondition : 사용자가 로그인 상태, 프로필을 생성한 상태 | | |
| TEST STEPS | | |
| # | STEP | Expected Result |
| 1 | 사용자가 프로필 화면에서 검색을 한다. | 검색된 사용자 결과를 return |
| 2 | 사용자가 특정 사용자에게 초대 메시지를 보낸다. | 해당 사용자에게 초대 메시지를 보낸다. |
| 3 | 초대를 받은 사용자가 수락한다. | 자동으로 교환일기가 생성된다. |
| TEST INFORMATION | | |
| Test Case Number : T\_05 | | |
| Functionality Tested : 교환일기 작성 초대 | | |
| Test Type : Negative | | |
| Precondition : 사용자가 로그인 상태, 프로필을 생성한 상태 | | |
| TEST STEPS | | |
| # | STEP | Expected Result |
| 1 | 사용자가 프로필 화면에서 검색을 한다. | 검색된 사용자 결과를 return |
| 2 | 사용자가 특정 사용자에게 초대 메시지를 보낸다. | 해당 사용자에게 초대 메시지를 보낸다. |
| 3 | 초대를 받은 사용자가 거절한다. | 거절 메시지를 사용자에게 출력해준다. |
| TEST INFORMATION | | |
| Test Case Number : T\_06 | | |
| Functionality Tested : 교환일기 작성 | | |
| Test Type : Positive | | |
| Precondition : 사용자가 로그인 상태, 프로필을 생성한 상태, 교환일기가 생성된 상태 | | |
| TEST STEPS | | |
| # | STEP | Expected Result |
| 1 | 사용자가 교환일기 작성 화면으로 이동한다. | 사용자에게 교환일기 작성페이지를 return |
| 2 | 사용자가 해당 폼에 맞게 교환일기를 작성하고 생성버튼을 누른다. | 작성된 일기에 작성한 사용자와 같은 소유권의 사용자가 일기를 확인할 수 있도록 출력한다. |
| TEST INFORMATION | | |
| Test Case Number : T\_07 | | |
| Functionality Tested : 교환일기 작성 | | |
| Test Type : Negative | | |
| Precondition : 사용자가 로그인 상태, 프로필을 생성한 상태, 교환일기가 생성된 상태 | | |
| TEST STEPS | | |
| # | STEP | Expected Result |
| 1 | 사용자가 소유하지 않은 교환일기 작성화면으로 이동한다. | 사용자에게 해당 교환일기에 대한 권한이 없다는 메시지를 출력한다. |

### 11.4.5. 시각화

|  |  |  |
| --- | --- | --- |
| TEST INFORMATION | | |
| Test Case Number : T\_08 | | |
| Functionality Tested : 시각화 자료 출력 | | |
| Test Type : Positive | | |
| Precondition : 사용자가 로그인 상태, 프로필을 생성한 상태, 일기를 몇 가지 작성한 상태 | | |
| TEST STEPS | | |
| # | STEP | Expected Result |
| 1 | 사용자가 프로필 화면으로 이동한다. | 사용자에게 프로필 화면을 출력한다. |
| 2 | 사용자가 시각화 분석 버튼을 클릭한다. | 해당 사용자에게 시각화 분석 결과를 출력한다. |
| TEST INFORMATION | | |
| Test Case Number : T\_09 | | |
| Functionality Tested : 시각화 자료 출력 | | |
| Test Type : Negative | | |
| Precondition : 사용자가 로그인 상태, 프로필을 생성한 상태, 일기를 작성하지 않은 상태 | | |
| TEST STEPS | | |
| # | STEP | Expected Result |
| 1 | 사용자가 프로필 화면으로 이동한다. | 사용자에게 프로필 화면을 출력한다. |
| 2 | 사용자가 시각화 분석 버튼을 클릭한다. | 해당 사용자에게 시각화 분석 결과가 없다는 메시지를 출력한다. |

## 11.5. Test Results

|  |  |  |  |
| --- | --- | --- | --- |
| TEST CASE NUMBER | Test Case Description | Test Case Test Result | Comments |
| T\_01 | 다이어리 선택작성(P) | Success |  |
| T\_02 | 다이어리 직접작성(P) | Success |  |
| T\_03 | 다이어리 직접작성(N) | Success |  |
| T\_04 | 교환일기 작성 초대(P) | Success |  |
| T\_05 | 교환일기 작성 초대(N) | Success |  |
| T\_06 | 교환일기 작성(P) | Success |  |
| T\_07 | 교환일기 작성(N) | Success |  |
| T\_08 | 시각화 자료 출력(P) | Success |  |
| T\_09 | 시각화 자료 출력(N) | Success |  |

## 11.6. Requirements Satisfaction

### 11.6.1. User Requirements

|  |  |
| --- | --- |
| No. | USER\_RS\_001 |
| Title | 다이어리 선택작성 |
| Test Unit | T\_01 |
| No. | USER\_RS\_002 |
| Title | 다이어리 직접작성 |
| Test Unit | T\_02, T\_03 |
| No. | USER\_RS\_004 |
| Title | 교환일기 |
| Test Unit | T\_04, T\_05, T\_06, T\_07 |
| No. | USER\_RS\_005 |
| Title | 시각화 |
| Test Unit | T\_08, T\_09 |

### 11.6.2. Non-Functional Requirement

|  |  |
| --- | --- |
| No. | NF\_RS\_001 |
| Title | Performance Requirement |
| Test Unit | T\_01, T\_02, T\_03, T\_04, T\_05, T\_06, T\_07, T\_08, T\_09 |
| No. | NF\_RS\_002 |
| Title | Usability Requirement |
| Test Unit | T\_01, T\_02, T\_03 |
| No. | NF\_RS\_003 |
| Title | Security Requirement |
| Test Unit | T\_06, T\_07 |
| No. | NF\_RS\_004 |
| Title | Operational Requirement |
| Test Unit | - |
| No. | NF\_RS\_005 |
| Title | Ethical Requirement |
| Test Unit | T\_06, T\_07 |

# Appendix

## A. Glossary

1. 반응형 웹: 크기가 고정된 페이지가 아닌 출력되는 디바이스의 크기에 따라 인터페이스가 바뀌는 웹페이지

2. SQL Query: 데이터베이스에 접근하기 위해서 사용되는 언어가 SQL을 의미하고 특정한 데이터를 조회 또는 수정을 위해 Query를 이용한다.

3. HTML: 대표적인 마크업 언어로 태그로 이루어진 언어다. 웹 페이지를 작성할 때 많이 사용하는 언어

4. UNIQUE KEY: 관계형 데이터베이스에서 데이터간 중복이 되지 않은 칼럼을 지칭하는 용어

5. 워드 클라우드: 주로 데이터 시각화에 많이 사용되는 방법으로 단어의 빈도수에 따라 크기를 크게 한 단어 뭉치

6. Entity: 개체를 의미하는 언어로 이 문서에서는 교환일기 개체 하나를 의미한다.

7. 파이차트: 점유율을 보여주기 유용한 시각화 방법 중 하나이다.

8. 인코딩: 사용자가 입력한 문자나 기호 등을 컴퓨터가 이용할 수 있는 신호로 만드는 것을 말한다.

9. Pk: Primary key를 의미하며 관계형 데이터베이스에서 데이터에 부여되는 고유의 값이다.

10. 페이지 렌더링: 서버로부터 HTML 파일을 받아 사용자에게 출력하는 과정을 의미한다.

11. Http: world wide web에서 정보를 주고받을 수 있는 프로토콜이다.

12. Request: 클라이언트가 웹 서버 측에 요청하는 것을 의미한다.

## B. Domain class Description

|  |  |
| --- | --- |
| class name | Cross\_Diary |
| description | make sure user is user of cross\_diary |
| relation | association to Profile |
| attribute | Name: string , User\_A: string, User\_B : string |

|  |  |
| --- | --- |
| class name | Cross\_Diary\_Content |
| description | store user’s cross\_diary content |
| relation | association to Cross\_Diary |
| attribute | Title : string, Contents : string, Create\_At : string, Update\_At : string , Name : string |

|  |  |
| --- | --- |
| class name | Event\_List |
| description | store user’s daily events |
| relation | association to Diary\_Content |
| attribute | Feeling: string |

|  |  |
| --- | --- |
| class name | Feeling\_List |
| description | store user’s daily feelings |
| relation | association to Diary\_Content |
| attribute | Feeling: string |

|  |  |
| --- | --- |
| class name | Diary |
| description | finding diary information from database |
| relation | association to user, association to Diary\_Content |
| attribute | User: string , Name : string |

|  |  |
| --- | --- |
| class name | Diary\_Content |
| description | User’s Diary information |
| relation | association to Feeling\_List, association to Event\_List, association to Diary |
| attribute | Title : string, Contents : string, Create\_At : string, Update\_At : string , Name : string |

|  |  |
| --- | --- |
| class name | Profile |
| description | User’s information |
| relation | association to user, association to cross\_diary, association to\_do\_list |
| attribute | Nickname :string , Image:Image, User:string |

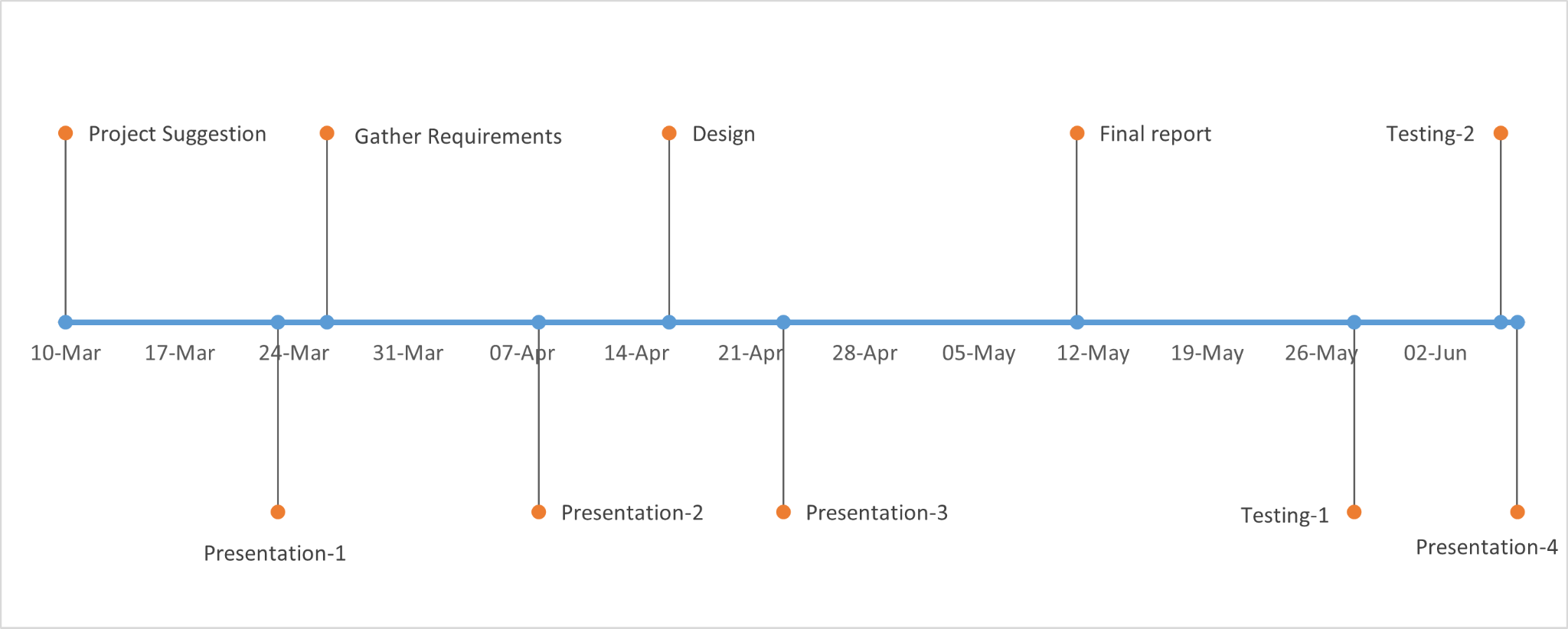
|  |  |
| --- | --- |
| class name | Todo\_List |
| description | Provide list of what user have to do |
| relation | association to profile |
| attribute | Nickname :string |

|  |  |
| --- | --- |
| class name | User |
| description | users with platform |
| relation | association to profile, association to diary |
| attribute | ID:string, Password:string, E-mail:string |

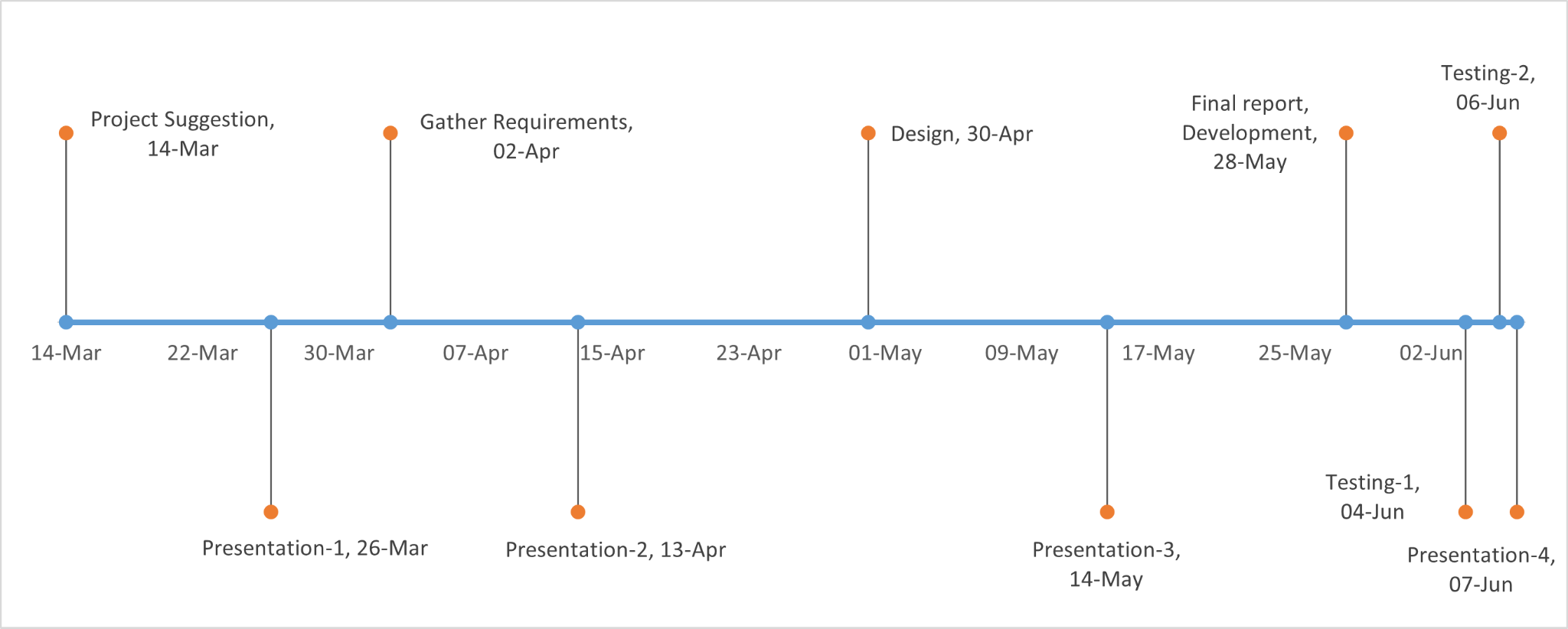
## 

## C. Project Status

### C.1 Planned vs. actual



<그림 36. Planned Project Status>



<그림 37. Actual Project Status>

### C.2 Team members’ contribution

용희원 18%

김민경 18%

강세빈 18%

허창현 18%

여찬복 28%

## E. Change History

|  |  |
| --- | --- |
| Previous Version | Project\_proposal\_001 |
| New Version | Requirement\_specification\_001 |
| Changes | 핵심 기능 1개 → 4개  ▷ 일기 선택작성 기능  ▷ **TODO list 작성 기능**  ▷ **교환일기 기능**  ▷ **일기 관련 통계자료 제공** |
| Reason | 핵심 기능 하나를 개발하는데 한 학기 동안 5명의 팀이 만들어진다는 것은 목표를 너무 낮게 잡은 경향이 있기 때문이다. 소프트웨어 공학 수업에서 배운 다양한 프로세스를 적용하기 위해서 핵심 기능을 추가했다. |

|  |  |
| --- | --- |
| Previous Version | Requirement\_specification\_001 |
| New Version | Design\_Document\_001 |
| Changes | 핵심 기능 4개 → 5개  ▷ **일기 직접작성 기능**  ▷ 일기 선택작성 기능  ▷ TODO list 작성 기능  ▷ 교환일기 기능  ▷ 일기 관련 통계자료 제공 |
| Reason | 일기 작성에 자유도를 높이기 위해서, 선택작성 기능 외에 직접작성 기능을 추가하였다. |

|  |  |
| --- | --- |
| Previous Version | Requirement\_specification\_001 |
| New Version | Design\_Document\_001 |
| Changes | USER\_RS별로 Activity Diagram 추가 |
| Reason | 프로세스 과정에 직관성을 더하기 위하여 USER\_RS별로 Activity Diagram을 추가하여 진행 과정을 잘 보이도록 하였다. |

## F. References

1. 액티비티 다이어그램 : <https://online.visual-paradigm.com/app/diagrams/#diagram:proj=0&type=ActivityDiagram>
2. OTO-System Architecture : <https://www.researchgate.net/figure/OTO-system-architecture-OTO-employs-client-server-architecture-and-utilizes-the_fig6_259539890>
3. 시퀀스 다이어그램 : <https://sequencediagram.org/>
4. 클래스 다이어그램 : <https://online.visual-paradigm.com/diagrams.jsp#diagramlist:proj=0&new=ClassDiagram>
5. 참조 블로그: <https://velog.io/@namezin/%EB%B0%98%EC%9D%91%ED%98%95-%EB%94%94%EC%9E%90%EC%9D%B8-%EB%8F%84%EC%9E%85%ED%95%98%EA%B8%B0-3.-%EC%82%AC%EC%9D%B4%ED%8A%B8-%EC%86%8D%EB%8F%84-%EB%86%92%EC%9D%B4%EA%B8%B0>

1. 알바몬 설문조사 자료 인용 [↑](#footnote-ref-1)
2. 경희대 정신건강의학과 백종우 교수 인터뷰 인용 [↑](#footnote-ref-2)
3. 블로그 참조 [↑](#footnote-ref-3)