“오픈소스SW개발” 팀프로젝트

**‘시나브로’ 최종 보고서**

**<팀 소개>**

|  |  |  |
| --- | --- | --- |
| **Team name** | **Team members** | **Role** |
| **growing** | **소프트웨어학부 2019036380 이다은** | Frontend, 일지작성, 최종보고서 |
| **소프트웨어학부 2019056462 임단비** | Frontend, 일지작성, Data Base |
| **소프트웨어학부 2019089034 김현진** | Frontend, 일지작성, Data Base |

**<프로젝트 소개>**

저희는 현대인들이 바쁜 삶 속에서 자신을 좀 더 들여다보고, 여러 긍정적인 변화를 도모할 수 있도록 다양한 기능을 넣어 Application을 개발할 예정입니다.

Application의 이름은 “시나브로”로 정했으며, 이는 "시나브로"의 의미(모르는 사이에 조금씩)처럼 앱을 사용하면서 차곡차곡 긍정적인 변화를 만들어 내기를 바라는 마음을 담았습니다.

또한, “시나브로”가 순 우리말인 만큼, Application의 전체 디자인을 되도록 순 우리말로 할 계획이며, 이를 통해 Application 사용자들은 더 많은 순 우리말을 접하고, 동시에 그 단어가 주는 따뜻함을 느낄 수 있게 될 것입니다.

# 프로젝트 Repository 주소 : <https://github.com/LIMDANBI/growing>

**<팀원들 간의 소통 방법>**

* **Slack**

교수님께서 소개해주신 Slack 어플을 활용하여 서로 개발 과정에 필요한 사이트 링크를 주고받으며 소통하였습니다. 교수님과 함께 채팅방에 들어가 있어 개발 과정에 대해 조언을 받을 수 있다는 장점이 있었습니다. 또한 원하는 날짜를 클릭하면 해당 날짜의 회의 내용을 바로 볼 수 있어서 개발의 흐름을 쉽게 파악할 수 있었습니다.

* **Kakao Talk**

팀원들이 서로 알고 있는 사이였기에 카카오톡 단체 채팅방을 통해 주로 소통하였습니다. 메시지로 소통을 하는 것 보다 빠르게 소통하기 위하여 직접 그룹콜을 하여 바로바로 피드백 하는 방식으로 진행하였습니다. 이를 통해 개발과정에서 막히는 부분을 바로 물어보고 서로 의견을 주고받는 방식으로 개발을 진행할 수 있었습니다. 통화내용은 기록할 수 없기에 그동안 해왔던 통화기록을 첨부하였습니다!

|  |  |  |  |
| --- | --- | --- | --- |
| 회의 날짜 | 그룹 콜 | 회의 날짜 | 그룹 콜 |
| 5월 11일 |  | 6월 3일 |  |
| 5월 13일 |  | 6월 23일 |  |
| 5월 17일 |  | 6월 24일 |  |
| 5월 24일 |  | 6월 25일 |  |
| 5월 30일 |  |  |  |

* **Github - Issue**

**#First Issue**

메인 화면이 나오기 직전에 뜨는 시작화면을 각자 디자인해보고 의견을 종합하여 시작화면을 디자인하였습니다.

**<초안>**

|  |  |  |
| --- | --- | --- |
| 임단비 | 김현진 | 이다은 |
| image | 5 | image |

**<완성된 디자인>**

|  |  |
| --- | --- |
| 배경화면4 | **최종 완성된 시작화면 디자인**  전체적인 색깔을 단비의 초안을 기본으로 삼고, 행복을 찾는 퍼즐을 맞춰 나간다는 의미를 담은 현진이의 시나브로 퍼즐디자인을 중심으로 잡았습니다. 마지막으로 ‘시나브로’의 뜻인 '모르는 사이에 조금씩 조금씩'을 보고 한 발자국 앞으로 나아가는 듯한 느낌을 받아서 우리의 발자국이라고 표현해 보았습니다. |

**#Second Issue**

각자 맡은 기능의 디자인을 공유하고 구현 방법을 알려 주는 과정을 통해 서로 피드백을 해주면서 깔끔하게 수정하였습니다.

**<한 줄 기록>**

|  |  |  |
| --- | --- | --- |
| **초안** | **날짜함수 추가** | **최종 구현 완료** |
| 스크린샷 2020-05-25 오후 1 38 00 | 한줄 기록 구현 |  |

**<소망 목록>**

|  |  |  |
| --- | --- | --- |
| **초안** | **입력 받는 기능 추가** | **최종 구현 완료 (버튼수정)** |
| image | image |  |

**<개발 진행과정>**

* **2020.05.11 개발방향 잡기** (Slack, Kakao Talk 그룹 콜 활용)

첫 회의를 하였습니다. 우선 팀프로젝트의 개발 방향을 잡기 위한 토의 결과 어플리케이션을 만들기로 하였고, 현대인에게 필요한 어플을 만들고자 여러 가지 의견을 종합한 결과 현대인의 스트레스를 관리해줄 수 있는 어플을 개발하기로 결정하였습니다.

* **2020.05.13 어플리케이션 기능 회의** (Slack, Kakao Talk 그룹 콜 활용)

메인 화면의 디자인을 어떻게 할지 회의를 진행한 결과 기능들을 퍼즐 모양으로 연결시켜주고 화면의 상단에 목표를 적을 수 있는 공간과 날짜 디데이기능을 보여줄 수 있도록 디자인하기로 결정하였습니다. 1차 초안과 2차 초안을 거쳐 최종 디자인을 구성하였습니다.

|  |  |  |
| --- | --- | --- |
| **1차 초안** | **2차 초안** | **최종 화면** |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 |  |  |

현대인들의 스트레스를 관리해주는 어플인 ‘시나브로’ 안에 들어갈 기능들을 생각해보는 시간을 가졌습니다. 회의 결과 세부적인 기능후보로 7개의 후보를 정하게 되었습니다.

*<세부적인 기능>*

* + - 1. 나만의 동화책

자신의 하루 하루를 그림으로 기록할 수 있고, 사용자의 한달 그림 기록을 매달 첫날 동화책처럼 엮어 사용자에게 제공해줍니다! (수고한 한달을 돌아보는 시간을 가지고 자신을 칭찬해주세요!, 당신은 한 달 동안 이런 일 때문에 웃고 슬펐군요! 이번 달은 더 행복한 한달이 될 거예요!)

1. 한 줄 기록

그날 그날의 감정을 짧게 기록하며, 스스로에게 더 귀 기울이는 시간을 가질 수 있습니다! (한 줄 한 줄 모인 글을 나중에 보면 그 또한 소중한 보물이 될 것입니다!)

1. 습관 다듬기

개선하고 싶은 습관, 만들고 싶은 습관들을 기록하고 이를 다듬어 나갈 수 있습니다! 오늘 하루도 당신은 더 좋은 사람이 되어가고 있습니다!

1. 소망 목록

한글판 ‘버킷 리스트’, 자신이 10대별로 해보고 싶은 일을 기록하고 이뤄 나갈 때 마다 체크하는 기능입니다 ! 한번뿐인 삶 실패를 두려워하지도, 하고 싶은 걸 주저하지도 마세요! 소망 목록을 채우고 모두 이뤄 나가세요!

1. 감상 책

영화, 드라마, 음악 등 예술, 문화 작품을 보거나 듣고 감상 평을 남기는 공간입니다. 요새 많은 사람들이 자신의 감상을 기록하는데, 이를 앱의 한 기능으로 추가하여 사용자에게 더 편리함을 줄 수 있습니다.

1. 탈출구

당신만의 스트레스 해소 방법, 우울함 극복 방법은 무엇인가요? 이곳에 보물처럼 넣어 놓으세요! (예를 들어, 유튜브를 즐겨본다 – 링크를 걸어 둘 수 있다 등) 그리고는 힘든 날에는 이 탈출구를 찾아보는게 어떨까요? 당신은 분명 행복해질 거예요!

1. 감정 쓰레기통

오늘 하루 자신을 힘들게 만들었던, 남에게는 하지 못할 말들을 쏟아내세요! 당신의 버리고 싶은 감정들을 감정 쓰레기통이 받아 줄 것입니다! 그리고는 함께 태워버립시다! (태우기 기능 제공) 여러분을 슬프게 했던 모든 일들이 사라졌습니다! 내일은 행복한 일들이 더 많았으면 좋겠네요!

*<부수적인 기능>*

* 1. 목표 설정

(당신은 지금 무엇을 향해 달려가고 있나요?) : 현재 자신의 목표를 상단의 고정하여 삶의 의욕을 더 증진시킬 수 있습니다.

* 1. D-day

목표 옆에 디데이 기능을 추가합니다. 목표를 이루기까지 남은 날들을 직관적으로 볼 수 있습니다.

* 1. 감정 기록 (행복 지수 / 슬픔 지수)

그 날 그 날의 감정을 쉽게 버튼만으로도 기록할 수 있습니다!

* **2020.05.17. 어플리케이션 구현할 기능결정** (Slack, Kakao Talk 그룹 콜 활용)

이 전 회의 결과로 나왔던 후보 중 구현할 기능 4개를 선정하는 시간을 가졌습니다. 최종 선정된 기능은 아래 4가지입니다.

1. 소망목록(버킷 리스트) : 이루고 싶은 일을 리스트로 보기 쉽게 정리하고 체크할 수 있는 칸도 추가하여 이뤄진 소망은 체크하는 기능.
2. 감정쓰레기통 : 힘들었던 일이나 하고싶지만 못했던 말들을 마음껏 쏟아내는 장소로, 리스트에 적어 두고 드래그 하여 쓰레기통에 버림으로써 속상한 마음을 속 시원히 털어내자는 취지를 담은 기능.
3. 한 줄 기록 : 하루의 감정을 한 줄로 짧게 기록하여 날짜 별로 나의 감정들을 살펴볼 수 있는 공간을 제공하는 기능.
4. 감상 책(리뷰노트) : 드라마나 영화 등의 예술작품을 보고 감상내용을 기록하는 공간을 제공하는 기능.

어플리케이션 개발 관련 지식이 부족하다는 것을 알고 각자 fragment와 Data Base에 관련한 공부를 해오기로 하였습니다.

* **2020.05.24. splash기능 구현완료, 어플리케이션 기능 분담** (그룹 콜 활용)
* 어플리케이션의 메인 화면을 띄우기 직전 첫 시작화면을 splash 기능을 사용하여 3초 간 띄워주는 기능을 구현 완료하였습니다. 또한 구현하고자 하는 4가지 기능의 구현 방법에 대해서 조사하고 감상책을 제외한 나머지 세 가지 기능을 한 명씩 맡아 구현해보기로 결정하였습니다. 메인 화면에서 각 기능들에 접근하는 방법을 토의한 결과 Fragment를 실행했을 때 activity를 사용했을 때보다 메모리 공간을 적게 차지하기 때문에 fragment로 구현하기로 결정하였습니다. 퍼즐모양을 이미지버튼으로 사용하여 클릭했을 때 각 기능에 맞는 fragment로 넘어갈 수 있게 구현하고자 합니다.

<기능 구현 방법 회의결과>

1. 소망목록(버킷 리스트) : Checkbox와 List View, Text View를 사용해서 구현할 예정.
2. 감정쓰레기통 : List View와 Text View, Drag and Drop을 사용해서 구현할 예정.
3. 한 줄 기록 : List View와 Text View를 사용해서 구현할 예정.
4. 감상 책 : List View, Image View, Text View를 사용해서 구현할 예정.

* **2020.05.30. fragment 구현, main activity에 띄울 날짜 함수 구현완료**

(Slack, Kakao Talk 그룹 콜 활용)

메인 화면을 구성하고 있는 4개의 퍼즐버튼을 구현하기 위해 이미지버튼 기능을 사용하였습니다. 각 버튼을 눌렀을 때 기능에 맞는 fragment로 이동하게 구현할 계획입니다. 또한 한 줄 기록에 사용되는 날짜 함수를 구현하여 메인 화면에도 띄워주는 작업을 하였습니다. 글씨체를 좀 더 깔끔하게 하기 위해 Font directory를 생성하고 나눔 명조체를 적용하여 구현하였습니다.

* **2020.05.31.** **한 줄 기록 구현완료, 나만의 목표 구현완료**

나만의 목표를 기록하면 그것을 메인 화면에 띄워주는 기능을 구현하였습니다. Edit Text를 사용하여 클릭하면 사용자가 글자를 쓸 수 있도록 구현하였습니다.

사용자의 입력을 받는 부분과, 받아서 List View를 통해 보여주도록 구현하였습니다. 또한 사용자를 격려하기 위해서 "결국 일상 속의 행복이다. 행복한 날은 아니었 어도, 하루 중 행복했던 일은 있다."라는 문구를 추가했습니다. 더 자세한 설명은 아래에 코드리뷰를 통해 설명하겠습니다.

|  |  |  |
| --- | --- | --- |
| **초안** | **피드백 - 날짜함수 추가** | **최종 구현 완료** |
| 스크린샷 2020-05-25 오후 1 38 00 | 한줄 기록 구현 |  |

* **2020.06.01. 버킷 리스트 기능 구현완료**

사용자의 소망을 입력 받아서 “추가”버튼을 통해 리스트에 넣어 주도록 구현하였습니다. 이뤄낸 소망을 체크하기 위해 체크박스를 리스트에 추가하였습니다. 여러 개를 다중으로 선택할 수 있게 multiple choice 방식을 사용하였습니다. 또한 리스트에서 체크한 것을 “삭제” 버튼을 통해 화면에서 지울 수 있도록 구현하였습니다. 더 자세한 설명은 아래에 코드리뷰를 통해 설명하도록 하겠습니다.

|  |  |  |
| --- | --- | --- |
| **초안** | **피드백 - 입력 받는 기능 추가** | **최종 구현 완료 (버튼수정)** |
| image | image |  |

* **2020.06.03. D-day 기능 구현완료** (Slack, Kakao Talk 그룹 콜 활용)

‘Day’ Text View를 누르면 달력창이 뜨고, 원하는 날짜를 선택하면 그에 맞는 디데이를 보여주도록 구현하였습니다. 처음 구현하였을 때 디데이버튼을 생성하여 버튼을 눌렀을 때 달력이 뜨는 방식으로 구현하였지만 팀원들 간의 피드백을 통해 버튼을 없애고 Text View 로 만들어서 날짜기능 아래에 띄워주도록 구현하여 깔끔하게 수정하였습니다.

|  |  |  |
| --- | --- | --- |
| **초안 (피드백 받기 전)** | **피드백 후 수정완료** | **최종(순우리말 수정)** |
|  |  |  |

또한 감상 책 기능구현을 시작하였습니다. Dialog를 통해 팝업창을 띄우고 팝업창에 감상을 기록하면 List View에 넣어주는 방식으로 기록할 예정입니다.

* **2020.06.23 감정쓰레기통 기능 구현완료**

드래그 앱 드롭 기능을 사용하여 나의 슬픈 마음을 쓰레기통에 버리는 모양으로 구현하고자 했지만 드래그 앤 드롭 기능의 사용에 어려움이 있었습니다. 팀원들과 회의한 결과 쓰레기통을 클릭하여 팝업창을 띄우고, 팝업창에 나의 슬픈 마음을 모두 쏟아낸 뒤, 그것을 쓰레기통에 버려버리는 모양으로 구현하기로 결정하였습니다. 이를 구현하기위해 AlertDialog.Builder 기능을 사용하였고 원래 드래그 하고자 했던 이미지를 팝업창의 setIcon으로 활용하였습니다. 더 자세한 설명은 아래에 코드리뷰를 통해 설명하도록 하겠습니다.

|  |  |  |
| --- | --- | --- |
| **초안 (피드백 받기 전)** | **피드백 후 수정완료** | **피드백 후 수정 완료(팝업)** |
|  |  |  |

* **2020.06.24 감상 책 기능 구현완료**

시험기간 전부터 계속 수정해오던 감상 책 기능 구현을 완료하였습니다. 감정쓰레기통의 팝업창과는 다르게 제목과 날짜, 내용 등 기록할 것이 많아 감상 책의 팝업기능은 Dialog 로 구현하였습니다. 새로운 xml 파일을 만들고 “나의 감상 추가하기” 버튼을 클릭하면 나의 감상을 기록할 수 있는 팝업창을 띄워줍니다. 그곳에 나의 감상을 입력하고 ‘저장’ 버튼을 클릭하면 List View에 추가되도록 구현하였습니다. 추가하기 버튼을 잘못클릭 하였을 때 나갈 수 있도록 ‘취소’ 버튼을 구현하여 팝업창을 내릴 수 있도록 기능을 추가하였습니다. 더 자세한 설명은 아래에 코드리뷰를 통해 설명하도록 하겠습니다.

<기능별 실행 방법>

* **나만의 목표 설정**

|  |  |
| --- | --- |
|  |  |
| 나만의 목표설정하기 Text를 누르면 사용자가 직접 Text를 입력할 수 있다. | “I can do it”을 입력하여 나만의 목표를 설정하였다. |

* **날짜, 디데이**

사용의 목표까지 오랜 시간이 남지 않았음을 응원, 격려하는 마음으로 영어 표현인 'D-day' 대신 '다직해야' 라는 순 우리말로 수정하였습니다.

\* 다직해야 : '기껏해야' 라는 뜻의 순 우리말입니다.

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 메인 화면에서 ‘다직해야’를 클릭한다. | 클릭하면 날짜를 선택할 수 있는 창이 뜬다. 내가 원하는 날짜를 선택한다. | 날짜를 설정하면 남은 기간을 알려준다. 이미 지나간 일은 +(기간)으로 표시된다. |

* **한줄기록**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 저장버튼 위에 있는 Edit Text를 클릭하고, 나의 하루를 기록한다. | 오늘의 한 줄을 기록하면 해당 날짜와 함께 나의 하루가 한 줄로 기록된다. | 그날의 기록을 좀 더 자세히 읽어보고 싶다면 해당 날짜를 클릭한다. 그러면 그 날의 기록을 더 자세히 읽어 볼 수 있다. |

* **소망목록**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| ‘당신의 소망을 기록합니다.’ 아래에 있는 Edit Text를 클릭하고 나의 소망을 기록한다. | 기록한 소망들은 체크박스와 함께 리스크에 저장된다. 버튼은 자유롭게 개수 상관없이 클릭할 수 있다. | 소망을 체크하고 ‘삭제’ 버튼을 누르면 체크한 소망들이 사라진다. 여러 개를 선택했다면 한번에 지워진다. |

* **감정쓰레기통**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 중간에 있는 쓰레기통 이미지를 클릭하면 감정을 쏟아낼 수 있는 팝업창이 뜬다. | 팝업창의 Edit Text에 내가 입어버리고 싶은 감정들을 마구 적는다. | 쏟아내고 싶은 감정들을 적은 후 버리기 버튼을 통해 쓰레기통에 버리고 슬픈 감정을 잊어버린다! |

* **감상 책**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| ‘나의 감상 추가하기’ 버튼을 클릭하면 나의 감상을 기록할 수 있는 팝업창이 뜬다. | 팝업창이 뜨면 hint에 적힌 것을 바탕으로 제목, 날짜, 내용을 기록한다. | 제목, 날짜, 내용 양식에 맞추어 나의 감상이 리스트에 저장되는 것을 확인할 수 있다. |

**<코드 리뷰>**

* **SplashActivity.java**

|  |
| --- |
|  |
| SplashActivity에서는 어플을 실행시켰을 때 첫 화면을 띄워주는 역할을 하는 코드입니다. 어플리케이션을 누르면 drawable에 저장해 두었던 splash 사진이 3000millis (3초) 간 보이며, 3초 후 메인 화면(MainActivity.java)로 넘어갑니다. |

* **MainActivity.java**

|  |
| --- |
|  |
| getDate()함수는 메인 화면에 년도.월.일 순서에 맞춰 오늘의 날짜를 구해주는 코드입니다. 내장된 Calendar를 사용하여 구현하였습니다.  이 함수를 통해 메인 화면에 오늘의 날짜를 보여주고 디데이기능을 구현하는 과정에서도 사용됩니다. |
|  |
| 메인화면에서 각 기능의 화면으로 이동시켜주기 위한 코드입니다. activity\_main.xml에서 만들어 두었던 이미지 버튼을 findViewById를 통해 불러와 새로운 변수로 선언하고 각 기능에 맞게 클릭(onClick)하면 각 기능 (한 줄 기록, 소망목록, 감정쓰레기통, 감상 책)의 화면으로 넘어가도록 구현하였습니다.  addToBackStack()을 통해 뒤로가기 버튼을 눌렀을 때 이전 화면으로 넘어가도록 구현하였습니다. |
|  |
| 메인 화면의 디데이 기능을 지원하는 코드입니다.  위에서 구현해 두었던 날짜 구하기 getData()함수를 사용하여 구현하였습니다. TextView가 onClick되면 Dialog로 달력을 띄워주고, 달력에는 오늘의 날짜에 체크가 되어 있습니다.  달력에서 내가 원하는 날짜를 클릭하고 확인버튼을 누르면 해당 날짜와 오늘의 날짜 간의 차이를 보여주도록 하였습니다.  getDday() 함수를 통해 조건문으로 오늘의 날짜보다 과거일때는 + (dday - today) 일 로 보여주도록 하였고, 미래일때는 - (dday - today) 일, 날짜가 오늘이라면 +0일을 출력하도록 구현하였습니다. |

* **activity\_main.xml**

|  |
| --- |
|  |
| ImageButton(@+id/single\_line\_btn)  : 한 줄 기록 기능으로 연결시켜주는 퍼즐모양의 이미지 버튼입니다.  ImageButton(@+id/ emotion\_trsh\_bth)  : 감정 쓰레기통 기능으로 연결시켜주는 퍼즐모양의 이미지 버튼입니다.  ImageButton(@+id/ bucketlist\_btn)  : 소망 목록 기능으로 연결시켜주는 퍼즐모양의 이미지 버튼입니다.  ImageButton(@+id/ rivew\_btn)  : 감상 책 기능으로 연결시켜주는 퍼즐모양의 이미지 버튼입니다.  TextView(@+id/today\_date)  : 메인 화면에 오늘의 날짜를 보여주기 위한 텍스트입니다.  EditText(@+id/goal)  : 나만의 목표를 설정하기 위한 공간으로 EditText 로 구현하여 사용자가 텍스트를 바꿀 수 있도록 하였습니다.  TextView(@+id/day)  : 메인 화면의 오늘의 날짜 아래 디데이를 설정하기 위해 사용되는 텍스트입니다. 순 우리말인 ‘다직해야’를 사용하여 텍스트를 설정해 두었습니다.  TextView(@+id/goalday)  : MainActivity에서 ‘다직해야’를 클릭하고 달력에서 설정한 날짜와 오늘의 날짜의 차이를 보여주는 텍스트입니다. 본래 아무런 텍스트로 가지고 있지 않지만 날짜가 설정되면 다직해야 옆에 입력 받은 텍스트를 띄워주도록 하였습니다. |

* **한줄기록**
* **SingleLineRecord.java**

|  |
| --- |
|  |
| MainActivity.java에 있는 날짜 구하기 함수와 동일한 코드입니다.  한 줄 기록을 입력했을 때 입력한 내용 옆에 기록했던 날짜를 저장하기 위해 선언하였습니다. |
|  |
| final View single\_line\_view = inflater.inflate(R.layout.fragment\_single\_line\_record, container, false)를 통해 xml로 정의된 view를 실제 객체화 시켜주었습니다.  ArrayList를 선언해두고 변수를 user\_record로 설정합니다. setOnClickListener를 통해 storage\_btn이 클릭되었을 때 실행내용을 구현해주었습니다.  ‘저장’ 버튼이 클릭되면 우선 구현해둔 날짜함수를 통해 오늘의 날짜를 출력하고 “ : ”를 출력한 뒤 EditText을 통해 사용자가 입력해 둔 기록을 출력합니다.  이를 user\_input\_data 로 선언하고 이전에 선언해 두었던 ArrayList 인 user\_record에 add()를 통해 리스트에 추가해주도록 구현하였습니다.  마지막으로 미리 선언해 두었던 adapter인 slistviewadapter에 notifyDataSetChanged() 함수를 사용하여 ListView의 항목들을 갱신하도록 하였습니다. |
|  |
| 저장해둔 한 줄 기록을 지우고 싶을 때 사용하기 위한 코드입니다.  setOnClickListener를 통해 del\_btn이 클릭되었을 때 실행내용을 구현해주었습니다.  ‘삭제’ 버튼을 클릭하면 single\_line\_list에 저장해둔 한 줄 기록들 중 체크해 둔 기록들을 리스트에서 지워주는 역할을 합니다.  현재 선택된 아이템들을 가져오기 위해 getCheckedItemPositions() 함수를 사용합니다.  체크된 내용을 삭제하고 난 뒤 clearChioces() 함수를 통해 모든 선택 상태를 초기화하여 주도록 하였습니다.  Storage\_btn 과 같은 방식으로 마지막에는 미리 선언해 두었던 adapter인 slistviewadapter에 notifyDataSetChanged() 함수를 사용하여 ListView의 항목들을 갱신하도록 하였습니다. |
|  |
| 기록해 둔 내용을 좀 더 자세히 읽어보고 싶을 때 사용되는 코드입니다.  List에 있는 item을 클릭하면 구현해 둔 Detail\_singleline.java 로 연결되도록 구현하였습니다.  이를 통해 리스트에 저장된 나의 한 줄 기록을 클릭하면 새로운 창으로 이동되어 나의 기록을 더욱 자세히 볼 수 있습니다. |
|  |
| 다중 선택을 하기 위하여 simple\_list\_item\_multiple\_choice라는 안드로이드 SDK에서 기본적으로 제공되는 Layout을 사용하였습니다. Adapter에 전달되는 Layout으로 "android.R.layout.simple\_list\_item\_multiple\_choice"를 선언하여 리스트에 저장되는 item이 TextView와 CheckBox 위젯이 함께 배치되도록 구현하였습니다. |

* **Detail\_singleline.java**

|  |
| --- |
|  |
| 사용자가 한 줄 기록을 한 후에 리스트에서 해당 내용을 클릭하면 보여지는 화면을 구성하는 코드입니다.  날짜를 가져와서 dayed로 선언하고 화면의 상단에 ‘당신의 dayed 일의 기록입니다.’ 라는 Text를 출력합니다.  또한 입력해 두었던 나의 기록은 화면의 중앙에 띄워주도록 하였습니다.  이를 통해 과거의 나의 기록을 더 자세히 읽을 수 있도록 하였습니다. |

* **fragment\_single\_line\_record.xml**

|  |
| --- |
|  |
| ImageView : 바탕화면이 되는 이미지를 띄워주는 역할을 합니다.  TextView(@+id/ order\_single\_line)  : 사용자에게 “오늘하루를 한 줄로 기록해보세요!” 라고 기능을 소개하는 텍스트입니다.  TextView(@+id/ encourage)  : 하루를 기록하는 사용자를 격려하기 위해 "결국 일상 속의 행복이다. 행복한 날은 아니었 어도, 하루 중 행복했던 일은 있다." 라는 격려문구를 띄워줍니다.  EditText(@+id/ single\_line\_record)  : 사용자가 그 날의 기억을 기록할 수 있게 입력을 받는 역할을 합니다.  Button (@+id/store)  : 사용자가 입력한 기록을 리스트에 추가해주는 역할을 수행하기 위한 버튼입니다.  Button (@+id/del)  : 사용자가 입력하여 리스트에 저장된 내용을 삭제하고 싶을 때 체크된 아이템들을 리스트에서 제거하는 역할을 수행하는 버튼입니다.  ListView((@+id/stored\_list)  : 사용자가 기록한 내용을 저장해두기 위한 list이다. |

* **activity\_single\_line\_detail.xml**

|  |
| --- |
|  |
| ImageView : 바탕화면이 되는 이미지를 띄워주는 역할을 합니다.  TextView(@+id/ detail\_main\_text)  : 새로운 창이 띄워졌을 때 언제의 기록인지 날짜 알려주기 위해 상단에 날짜를 고지해주는 역할을 수행하는 텍스트입니다.  TextView(@+id/ detailed)  : 새로운 창이 띄워졌을 때 내가 보고자 했던 기록을 중간에 자세히 띄워주는 역할을 하는 텍스트입니다. |

* **소망목록**
* **Bucketlist.java**

|  |
| --- |
|  |
| View bucket\_view = inflater.inflate(R.layout.fragment\_bucketlist, container, false)를 통해 xml로 정의된 view를 실제 객체화 시켜주었습니다.  ArrayList를 선언해두고 변수를 dream\_record로 설정합니다.  setOnClickListener를 통해 add\_btn이 클릭되었을 때 실행내용을 구현해주었습니다.  ‘추가’ 버튼이 클릭되면 EditText를 통해 사용자가 입력해 둔 기록을 출력합니다. 사용자의 입력을 user\_dream으로 선언하고, 이전에 선언해 두었던 ArrayList 인 dream\_record에 add()를 통해 추가해주도록 구현하였습니다.  마지막으로 미리 선언해 두었던 adapter인 bucketViewAdapter에 notifyDataSetChanged() 함수를 사용하여 ListView의 항목들을 갱신하도록 하였습니다. |
|  |
| 저장해둔 소망을 지우고 싶을 때 사용하기 위한 코드입니다.  setOnClickListener를 통해 delete\_btn이 클릭되었을 때 실행내용을 구현해주었습니다. ‘삭제’ 버튼을 클릭하면 bucket\_list에 저장해둔 소망들 중 체크해 둔 기록들을 리스트에서 지워주는 역할을 합니다.  현재 선택된 아이템들을 가져오기 위해 getCheckedItemPositions() 함수를 사용합니다.  체크된 내용을 삭제하고 난 뒤 clearChioces() 함수를 통해 모든 선택 상태를 초기화하여 주도록 하였습니다.  add\_btn 과 같은 방식으로 마지막에는 미리 선언해 두었던 adapter인 bucketViewadapter에 notifyDataSetChanged() 함수를 사용하여 ListView의 항목들을 갱신하도록 하였습니다. |

* **fragment\_bucketlist.xml**

|  |
| --- |
|  |
| ImageView : 바탕화면이 되는 이미지를 띄워주는 역할을 합니다.  TextView(@+id/ new\_dream)  : 사용자에게 “당신의 소망을 기록합니다.” 라고 기능을 소개하는 텍스트입니다.  EditText(@+id/ new\_record)  : 사용자가 자신의 소망을 기록할 수 있게 입력을 받는 역할을 합니다.  ListView((@+id/bucketlist)  : 사용자가 기록한 내용을 저장해두기 위한 list이다.  Button (@+id/add)  : 사용자가 입력한 기록을 리스트에 추가해주는 역할을 수행하기 위한 버튼이다.  Button (@+id/delete)  : 사용자가 입력하여 리스트에 저장된 내용을 삭제하고 싶을 때 체크된 아이템들을 리스트에서 제거하는 역할을 수행하는 버튼이다. |

* **감정쓰레기통**
* **EmotionalTrashCan.java**

|  |
| --- |
|  |
| View trash\_view = inflater.inflate(R.layout.fragment\_emotional\_trash\_can, container, false)를 통해 xml로 정의된 view를 실제 객체화 시켜주었습니다.  setOnClickListener를 통해 trash\_btn이 클릭되었을 때 실행내용을 구현해주었습니다. 이미지버튼인 trash\_btn을 누르면 팝업창이 뜨도록 구현하기 위해 AlertDialog.Builder emo = new AlertDialog.Builder(trash\_view.getContext())을 사용하여 Dialog emo를 만들어주었습니다.  팝업창을 구성하기 위해 setIcon() 을 통해 미리 저장해둔 이모티콘 이미지를 아이콘으로 설정하고, setTitle()을 이용하여 “감정쓰레기통”이라는 제목을 지정해주었습니다.  사용자에게 기능사용방법을 안내하기 위해 setMessage()로 “잊어버리고 싶은 감정들을 적어주세요”고 소개하였고, 사용자의 입력을 받을 수 있도록 Edit Text를 선언하였습니다. 적은 슬픈 감정들을 쓰레기통에 버리는 버리기 버튼을 만들어 사용자가 적은 내용을 버리는 과정을 팝업창이 없어지는 방식으로 구현하였습니다. 없어지는 것은 dialog.dismiss()를 통해 구현하였습니다.  쓰레기통 이미지버튼을 눌렀을 때 팝업창이 보일 수 있게 emo.show()를 통해 구현하였습니다. |

* **fragment\_emotional\_trash\_can.xml**

|  |
| --- |
|  |
| TextView  : 사용자에게 “오늘 하루 당신이 버리고 싶은 감정은 무엇인가요?” 라고 기능을 소개하는 텍스트입니다.  TextView(@+id/ text)  : “당신이 버리고 싶었던 감정을” 이라는 텍스트입니다.  TextView(@+id/ text2)  : 사용자에게 “쓰레기통에 버리세요.” 라고 기능을 소개하는 텍스트입니다.  ImageButton(@+id/ trash)  : 사용자가 슬픈 마음을 적을 수 있는 팝업창을 띄워 주기 위한 이미지버튼이다. 쓰레기통모양으로 생겼으며 사용자는 쓰레기통을 누르고 마음을 비워낸 뒤 버리는 방식으로 구현하고자 이미지버튼을 사용하였다. |

* **감상 책**
* **ReviewNote.java**

|  |
| --- |
|  |
| View view = inflater.inflate(R.layout.fragment\_review\_note, container, false)를 통해 xml로 정의된 view를 실제 객체화 시켜주었습니다.  ArrayList를 선언해두고 변수를 review\_record로 설정합니다.  setOnClickListener를 통해 r\_btn이 클릭되었을 때 실행내용을 구현해주었습니다. r\_btn을 누르면 팝업창이 뜨도록 구현하기 위해 final Dialog review\_input = new Dialog(getActivity())를 이용하여 팝업창이 뜨도록 만들었고, 팝업창의 내용을 구성하기 위해 미리 만들어 둔 review\_input.xml을 setContentView()로 연결시켜 주었습니다. |
|  |
| store.setOnClickListener()는 ‘나의 감상 추가하기’를 통해 사용자가 팝업창에 입력한 기록을 ‘저장’ 버튼을 통해 리스트 뷰에 넣어 주기 위한 코드입니다.  감상제목 EditText에 쓴 내용은 앞에 ‘제목 : ’ 과 함께 첫번째 줄에 출력합니다. 감상날짜 EditText에 쓴 내용은 앞에 ‘날짜 : ’ 와 함께 두번째 줄에 출력합니다 감상내용 EditText에 쓴 내용은 앞에 ‘내용 : ’ 와 함께 세번째 줄에 출력합니다.  세 줄이 하나의 item에 되어 review 로 선언하고 이전에 선언해 두었던 ArrayList 인 review\_record에 add()를 통해 추가해주도록 구현하였습니다.  미리 선언해 두었던 adapter인 rlistViewAdapter에 notifyDataSetChanged() 함수를 사용하여 ListView의 항목들을 갱신하도록 하였습니다.  팝업 창 내에 취소버튼을 누르면 팝업창이 내려가도록 구현하기 위해 cancel.setOnClickListener()의 내용을 review\_input.dismiss()를 사용하여 팝업창이 꺼지도록 구현하였습니다.  ‘나의 감상 추가하기’버튼을 눌렀을 때 팝업창이 보일 수 있게 review\_input.show()를 통해 구현하였습니다. |
|  |
| 리스트 뷰를 구성하기 위하여 simple\_list\_item\_1 안드로이드 SDK에서 기본적으로 제공되는 Layout을 사용하였습니다.  Adapter에 전달되는 Layout으로 "android.R.layout.simple\_list\_item\_1"를 선언하여 리스트에 TextView가 저장되도록 하였습니다. |

* **fragment\_review\_note.xml**

|  |
| --- |
|  |
| TextView(@+id/review\_mtext)  : 사용자에게 “당신만의 감상공간이에요, 차근히 채워 나가세요!” 이라고 기능을 소개하는 텍스트입니다.  TextView(@+id/review\_stext)  : “그 때의 나는 어떤 작품을 좋아했고,” 이라는 텍스트입니다.  TextView(@+id/review\_stex2)  : “어떤 생각을 가지고 있었는지 추억할 수 있을 거예요.” 이라는 텍스트입니다.  ListView((@+id/review\_list)  : 사용자가 기록한 내용을 저장해두기 위한 list입니다.  Button (@+id/review\_add\_btn)  : 사용자가 감상을 기록할 수 있도록 팝업창을 띄워주는 버튼입니다. “나의 감상 추가하기”버튼을 누르면 감상을 기록할 수 있는 공간인 팝업창이 나와 입력을 할 수 있게 됩니다. |

* **review\_input.xml**

|  |
| --- |
|  |
| TextView  : 사용자에게 “감상기록공간” 이라고 기능을 소개하는 텍스트입니다.  EditText(@+id/ review\_Dsubject)  : 사용자의 입력을 받는 공간으로 제목을 입력 받기 위해 hint로 ‘감상제목’을 적어 두었습니다.  EditText(@+id/ review\_Dday)  : 사용자의 입력을 받는 공간으로 날짜를 입력 받기 위해 hint로 ‘감상물의 날짜’를 적어 두었습니다.  EditText(@+id/ review\_Dreview)  : 사용자의 입력을 받는 공간으로 내용을 입력 받기 위해 hint로 ‘감상내용’을 적어 두었습니다.  Button (@+id/ review\_store)  : 사용자가 입력한 기록을 리스트에 추가해주는 역할을 수행하기 위한 버튼입니다.  Button (@+id/ review\_cancel)  : 사용자가 입력한 기록을 취소하고 싶을 때 누르는 버튼으로 ‘취소’버튼을 누르면 팝업창이 내려가고 이전 화면으로 돌아갑니다. |

**여기까지 Team growing의 오픈소스 개발 프로젝트 최종 보고서였습니다! 감사합니다!**

**<Data Base구현>**

**<데이터베이스 소개>**

* **SQLite**

SQLite은 어디서나 실행 가능하다는 장점이 있습니다. 또한 날씨 앱 과제를 통해 안드로이드 스튜디오에서 앱을 빌드하는 과정에서 provider과 Service를 사용하여 안드로이드 스튜디오 내에서 데이터베이스와 연동하는 방법을 학습하였습니다. 이를 이용하고자 저희도 어플리케이션 개발 팀프로젝트에서 SQLite를 활용하고자 합니다!

SQL Helper를 통해서 데이터베이스를 생성합니다.

SQLiteOpenHelper 클래스를 사용하여 어플이 시작되고 있는 동안이 아닌 필요한 경우에만 데이터베이스 생성 및 업데이트와 같은 잠재적으로 장시간 실행될 수 있는 작업을 수행합니다.

**<기능 별 데이터베이스 구현>**

* **MainActivity**
* **main\_DBHelper**

|  |
| --- |
|  |

* **MainContract**

|  |
| --- |
|  |

* **한 줄 기록**
* **single\_line\_DBHelper**

|  |
| --- |
|  |

* **SingleLineContract**

|  |
| --- |
|  |

* **SingleLineDTO**

|  |
| --- |
|  |

* **소망 목록**
* **bucket\_list\_DBHelper**

|  |
| --- |
|  |

* **BucketListContract**

|  |
| --- |
|  |

* **BucketListDTO**

|  |
| --- |
|  |

* **감상 책**
* **review\_note\_DBHelper**

|  |
| --- |
|  |

* **reviewnoteContract**

|  |
| --- |
|  |