소프트웨어프로젝트 1

프로젝트 2 어플리케이션 개발

과 목 소프트웨어프로젝트 1

담당교수 이우신교수님

학 과 컴퓨터정보공학부

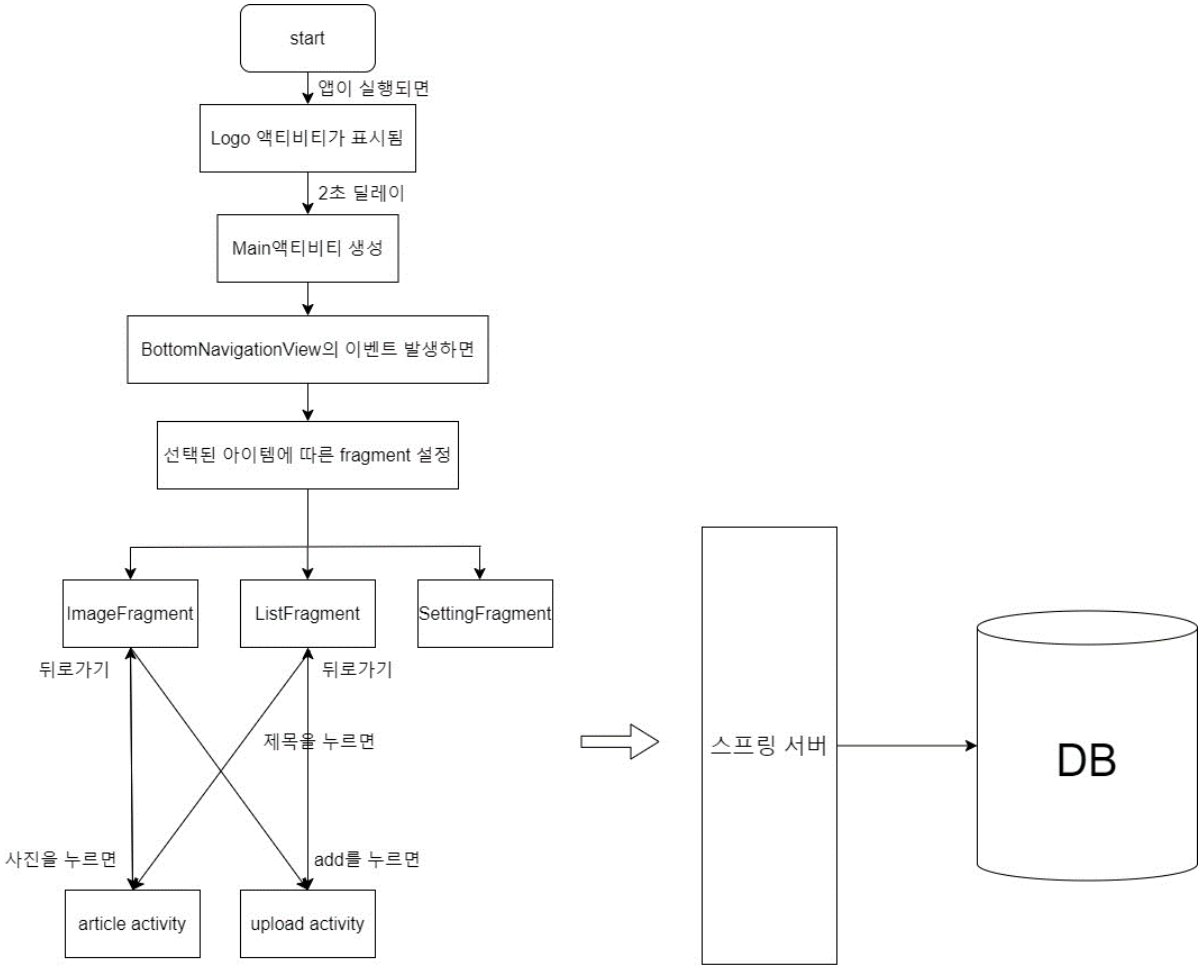
학 번 2021202058

이 름 송채영

[Introduction]

이번 프로젝트는 안드로이드 어플리케이션 개발을 구현하는 것이다. 프로젝트에서 MainActivity 는 하단 탐색 메뉴, BottomNavigationView 를 통해 3 가지 메인 화면, ImageFragment 그리드뷰, ListFragment 리스트뷰, settingFragment 으로 이동할 수 있다. 그리드뷰는 이미지를 표시하고 선택할 수 있으며, 리스트뷰는 글 목록을 표시한다. 마지막으로 settingFragment 는 어플리케이션 설정을 관리하지만, 이번 프로젝트에서는 따로 구현하지 않는다. 로고 액티비티는 실행 시 나타나는 첫 화면이다. 이어서 article 액티비티는 글의 내용을 자세히 보여주는 화면이다. 마지막으로 upload 액티비티는 이미지를 업로드하고 글을 작성하는 기능을 제공한다. 이번 프로젝트에서 위의 조건을 만족하는 코드와 다양한 기능과 화면 간의 상호작용을 구현하여 안드로이드 어플리케이션 개발에 대해 이해하고 실습해보며 그에 대한 기초를 학습해보았다.

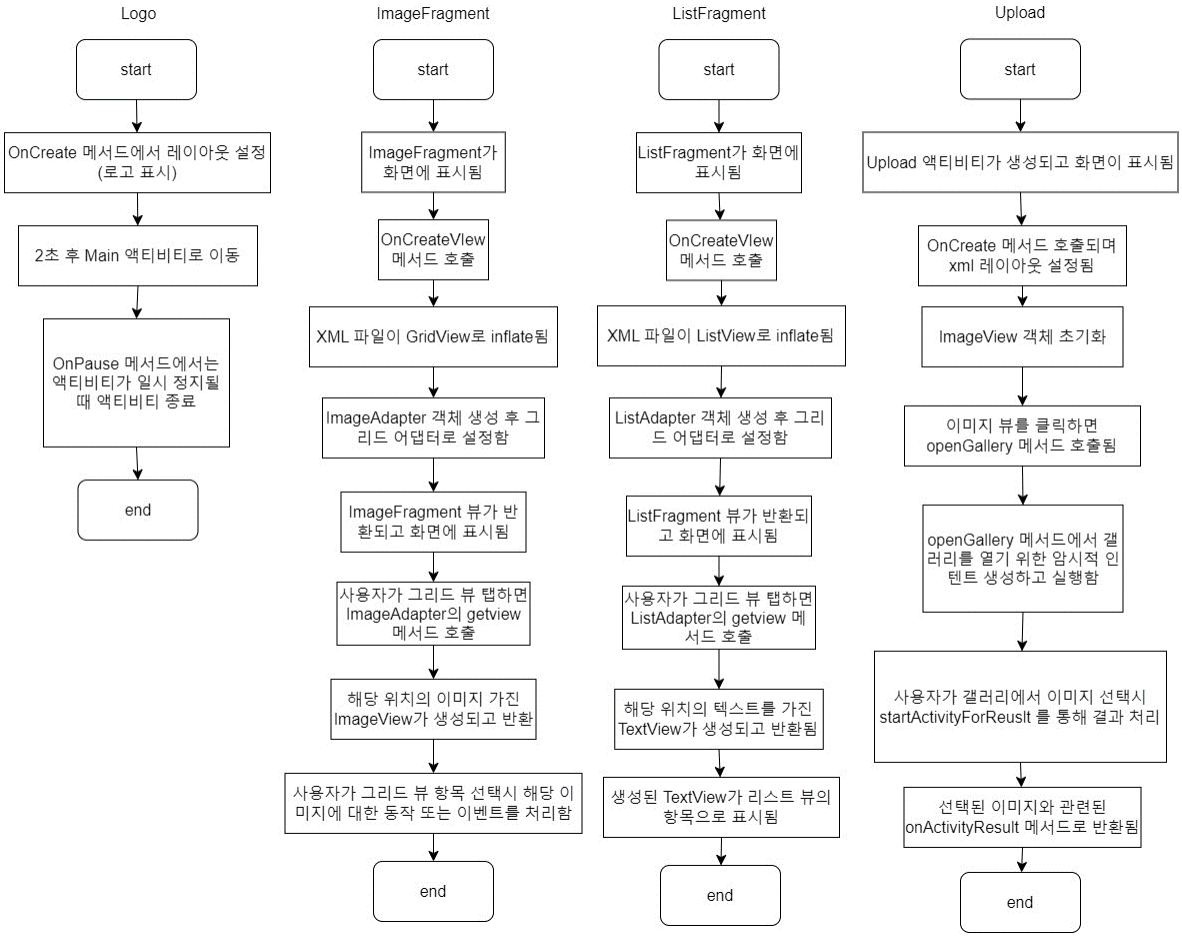
[Flow Chart]



과제 코드의 전반적인 흐름을 설명한 flow chart 이다. 앱이 실행되면 로고 액티비티가 표시된다. 로고 액티비티에서 2 초 동안 딜레이를 주고 그 후에 min 엑테비티로 이동하는 핸들러가 설정된다. main 액티비티가 설정되고 flowchart 에는 그리지 않았지만, onCreate

메서드에서 BottomNavigationView 객체를 초기화해서 아이템 선택 이벤트를 처리하는 리스너를 설정하였다. BottomNavigationView 의 아이템 선택 이벤트가 발생하면 BottomNavigate 메서드가 호출되며, 선택된 아이템에 따라 Fragment 를 설정한다. Gallery 를 선택 시 GridView 에 해당하는 ImageFragment 로, Title 을 선택하면 Listview 에 해당하는 TitleFragment 로, Setting 을 선택하면 SettingFragment 로 이동한다. 각각에 view 의 흐름에 대해서는 아래에서 설명하도록 하겠다. 먼저 ImageFragment 에서 사진을 누르면 Article 액티비티로 이동한다. Article 액티비티에서 뒤로가기를 누를 시 다시 그리드뷰로 이동한다. 그리드뷰에서 ADD 버튼을 누를 시 Upload 액티비티로 이동한다. 뒤로가기를 누르면 다시 그리드뷰로 이동한다. ListFragment 에서 글을 누르면 Article 액티비티로 이동하며 뒤로가기를 누를 시 리스트뷰로 이동한다. 리스트뷰에서 Add 를 누르면 upload 액티비티로 이동하며 여기서 뒤로가기를 누르면 리스트뷰로 이동한다. 이에 해당하는 응답을 서버로 보내어 데이터베이스에 최종적으로 저장하는 것이 목표이지만, 서버와 연결하는 부분을 코드로 구현하지 못해 해당 부분을 자세하게 그리지는 못하였다.

다음으로 각 view 의 흐름을 살펴보면 다음과 같다.



순서대로 logo activity, ImageFragment, ListFragment, Upload activity 이다. Article activity 는 서버와 연결하지 못했으므로 구현하지 못하였다. 구현 방식에 대해서는 Result 에서 설명하도록 하겠다. ImageFragment 는 Gallery 를 클릭했을 때, ListFragment 는 Title 을 클릭하였을 때 나타나는 화면으로 그에 해당하는 흐름을 나타내었다. logo activity 는 로고, upload activity 는 사진과 제목 그리고 글을 올릴 수 있는 view 에 해당한다.

[Result]

우선 프로젝트 내에 MVC Pattern 에 대해 설명하겠다. MVC 는 Model-View-Controller 의 약자로 소프트웨어 디자인 패턴을 의미한다. Model 과 View 그리고 Controller 로 나뉜다.
우선 Model 은 데이터와 비즈니스 로직을 담당하며 데이터의 상태를 유지하고 조작하는 역할을 수행한다. 데이터의 변경 사항을 view 와 controller 에게 알리는 역할을 한다. 코드에서의 Model 에는 Article 이 해당한다. Article 은 게시글의 데이터를 나타낸다.
다음으로 view 는 사용자 인터페이스(UI)를 담당한다. model 의 데이터를 시각적으로 표현하는 역할을 수행하며 사용자의 입력에 대한 이벤트를 controller 에게 전달한다. 코드에서의 View 에 해당하는 부분은 ImageFragment 와 ListFragment, Logo 그리고 XML 파일들이 있다. 먼저 ImageFragment 는 이미지를 그리드에 표시하는 fragment 의 view 역할을 하며 ListFragment 도 마찬가지로 텍스트를 리스트 형태로 표시하는 fragment 의 view 역할을 한다. Logo 는 로고 화면을 표시하는 액티비티의 view 역할을 하며 activity_article.xml, activity_logo.xml, activity_main.xml, acrivity_upload.xml, fragment_image.xml, fragment_list.xml, fragmet_setting.xml 에 해당하는 XML 파일들은 레이아웃과 UI 요소들을 정의하는 view 의 구조를 표현하므로 View 에 해당한다.
다음으로 controller 는 model 과 view 를 연결하고 상호 작용을 조정한다. 사용자의 입력을 받아 model 또는 view 에 반영하는 역할을 하며 model 의 상태 변화에 따라 view 를 업데이트 하고 view 의 변화에 따라 model 을 업데이트 한다. 코드에서 이에 해당하는 부분은 MainActivity, SettingFragment, Upload 가 있다. MainActivity 는 어플리케이션의 진입점이 되는 액티비티로, 뷰와 모델 간의 상호작용을 관리한다. settingFragment 는 설정을 관리하는 fragment 로 뷰와 모델간의 상호작용을 관리한다. 마지막으로 Upload 는 이미지 업로드를 담당하는 액티비티로 뷰와 모델간의 상호작용을 관리한다.

이어서 각 view 별 구현 방식에 대해 설명해보겠다.
우선 Logo 액티비티는 앱 실행 시 처음에 표시되는 로고화면을 담당하며 logo 의 구현방식은 다음과 같다. Oncreate 메서드에서 setContentView 를 호출해 XML 레이아웃(activity_logo)를 화면에 표시한다. Handler 객체를 생성해 2 초 후에 Runnable 객체를 실행한다. Runable 객체에서는 Intent 객체를 생성해 MainActivity 로 이동하며 startActivity 를 호출해 MainActivity 를 시작한다. finish 를 호출해 현재 액티비티인

logo 를 종료한다. OnPause 메서드는 액티비티가 일시 정지되면 호출되며, super.Onpause 를 호출해 기본 동작을 한 후 finish 를 호출해 액티비티를 종료한다. 흐름을 위의 flow chart 에서도 나타냈듯이 Oncreate 에서 로고화면을 표시한 후 2 초 뒤 MainActivity 로 이동한다. Onpause 에서 액티비티가 일시 정지될 때 액티비티를 종료한다.

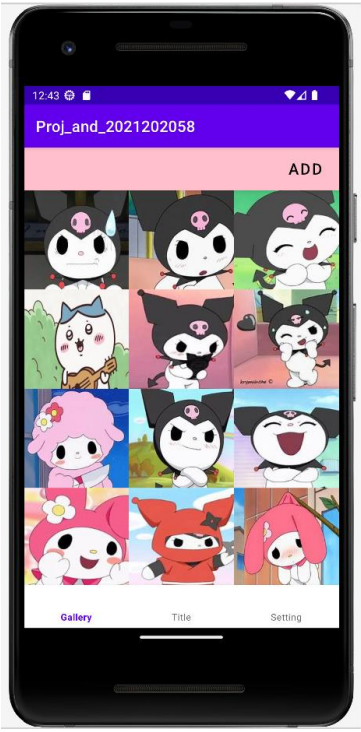
다음으로 ImageFragment 는 이미지를 그리드형태로 표시하는 fragment 로 구현방식은 다음과 같다. Oncreateview 메서드에서 xml 레이아웃을 fragment 에 연결한다. GridView 객체를 초기화하고 커스텀 그리드 어댑터의 인스턴스를 생성한다. 이어서 그리드 뷰에 어댑터를 설정한 후 그리드 뷰를 반환한다. ImageAdapter 클래는 BaseAdapter 를 상속한 커스텀 어댑터로 context 멤버 변수를 통해 어뎁터가 사용되는 컨텍스트를 저장하고 image 배열에 그리드에 표시할 이미지 리소스를 저장한다. getCount, getItem, getItemId 메서드를 통해 아이템 개수와 해당 위치의 아이템, 아이템 ID 를 반환한다. getView 메서드에서는 각 아이템의 뷰를 생성하여 반환한다. 이부분에서 이미지 뷰의 크기 등을 설정한다.

다음으로 ListFragment 는 텍스트를 리스트 형식으로 표시하는 fragment 로 구현방식은 다음과 같다. Oncreateview 메서드에서 xml 레이아웃을 fragment 에 연결한다. ListVuew 객체를 초기화하고 커스텀 리스트 어댑터의 인스턴스를 생성한다. 이어서 리스트 뷰에 어댑터를 설정한 후 리스트 뷰를 반환한다. ListAdapter 클래스는 BaseAdpater 를 상속한 커스텀 어댑터로 context 멤버 변수를 통해 어댑터가 사용되는 text 를 저장한다. text 배열에 리스트에 표시할 텍스트들을 저장한다. getcount, getItem, getItemID 메서드를 통해 아이템 개수, 해당 위치의 아이템, 아이템 ID 를 반환한다. getView 메서드에서는 각 아이템의 뷰를 생성해 반환한다. 이부분에서는 텍스트를 설정하고 텍스트 크기 등을 설정한다.

Article 액티비티는 구현하지 못했으므로 생략하였다. 마지막으로 Upload 액티비티는 이미지를 업로드하는 액티비티로 구현방식은 다음과 같다. setContentView 를 호출해 xml 레이아웃을 액티비티에 연결한다. setOnclickListener 를 사용해 이미지뷰를 클릭 시 openGallery 메서드를 호출한다. setOnclickListenerfmf 사용해 액티비티 종료 버튼을 클릭하면 finish 를 호출해 액티비티를 종료한다. openGallery 메서드에서는 Intent.ACTION_PICK 과 MediaStore.Images.Media.EXTERNAL_CONTENT_URI 를 사용하여 갤러리를 열기 위한 암시적 인텐트를 생성한다. startActivityForResult()를 호출하여 암시적 인텐트를 실행하고 결과를 처리한다.

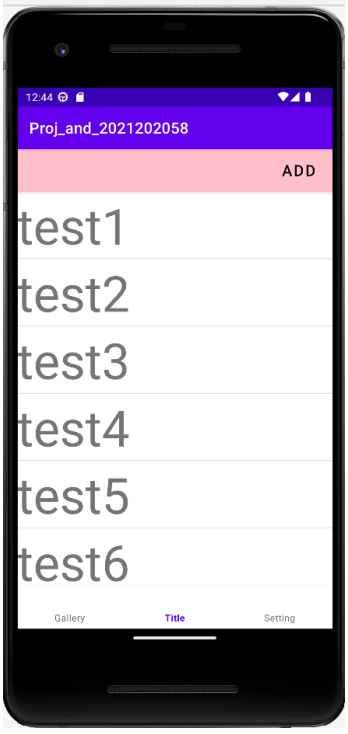
이어서 결과화면에 대해 설명하겠다.

로고를 보여주는 결과화면이다. 한 화면에 로고만 등장하며 2 초 후 그리드뷰로 이동한다. 뒤로가기를 눌러도 이동하지 않는 것을 확인하였다.

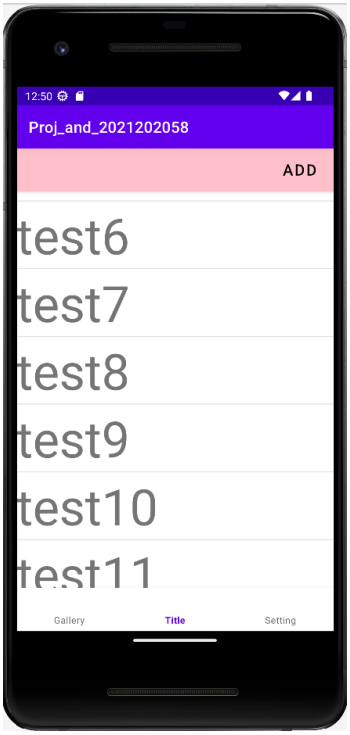


그리드뷰에 해당하며 Gallery 를 눌렀을 때 보여지는 화면이다. 가로 3, 세로 4 의 형태로 만들었다. 서버와 연결을 하지 못해 static 으로 사진을 직접 넣어 결과를 확인해보았다.

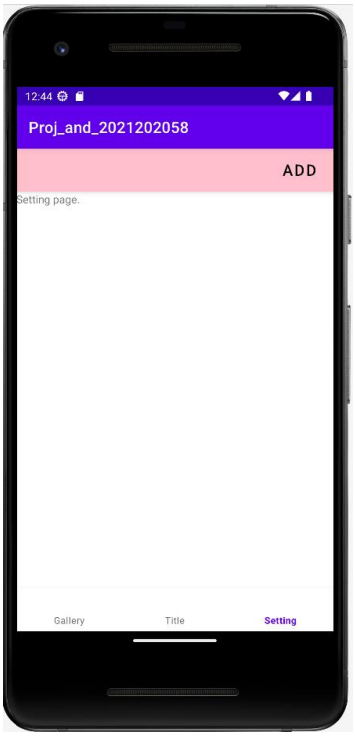
화면 스크롤을 통해 밑으로 내릴 수 있도록 하였으며 넣어둔 사진 모두 잘 나오는 것을 확인할 수 있다. 서버와 연결하지 못했으므로 최신순으로 정렬이 안되었다.



다음으로 title 을 눌렀을 때 보여지는 화면이다. 모든 글의 제목을 보여주어야 하지만, 서버와 연결하지 못했으므로 임시로 글을 넣어두어 결과를 확인하였다. 가로로 1 개, 세로로 6 개가 한 화면에 나오는 것을 확인할 수 있다.



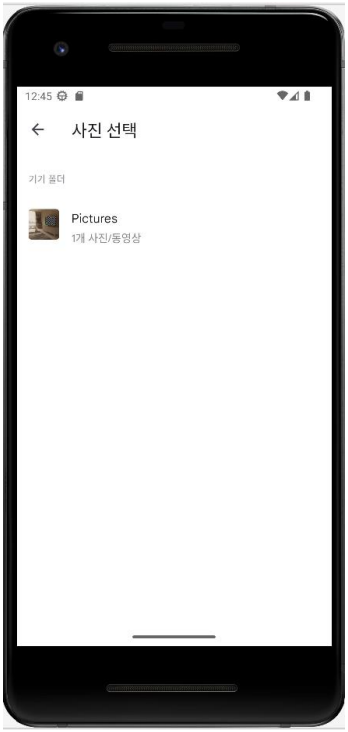
또한 스크롤을 통해서 아래의 글까지 볼 수 있도록 하였다.



다음으로 setting 을 눌렀을 때 보이는 화면이다. 이번 프로젝트에서 setting 은 별도로 구현하지 않아도 되기 때문에 tab 으로 전환되는 것만 확인하였다.

글이나 사진을 눌렀을 때 시각화 하는 article 은 서버와 연결하지 못했으므로 구현이 불가능한 부분이라 생각되어 결과화면 역시 없다. 따라서 수정 삭제와 관련된 부분도 구현하지 못하였다.

그리드뷰나 리스트뷰에서 ADD 버튼을 눌렀을 때 나타나는 화면이다. 사진과 재목 그리고 글을 올릴 수 있는 upload view 로 뒤로가기를 누르면 해당 뷰로 돌아간다. 사진과 글을 입력할 수 있는 칸과 업로드 버튼을 구현은 하였지만, 작동은 하지 않는다. 이 역시 서버와 관련된 부분이라 생각됐기 때문이다.



위의 사진에서 카메라 버튼, 즉 사진을 업로드 하기 위한 아이콘을 누르면 갤러리가 나타나는 것을 볼 수 있다. 사진을 선택할 수 있는 기능에 해당한다.

[Consideration]

이번 과제는 1 차 과제를 바탕으로 진행하는 과제였다. 다만 1 차 과제에서 진행했던 스프링 코드를 바탕으로 서버와 연결해서 데이터베이스에 저장해야 했지만, 서버와 연결하는 부분을 구현하지 못해 안드로이드 코드에서도 서버와 관련된 부분을 구현하지 못해 데이터베이스에 최종적으로 저장하지 못하였다. 서버와 연결하지 못해 그리드뷰에서 사진을 클릭하는 부분과 연관된 코드, article 도 구현하지 못하였고, 따라서 수정과 삭제 역시 구현하지 못했다. 또한 add 를 눌러 글을 작성할 수 있는 Upload 에서도 갤러리를 눌러 사진을 선택할 수는 있으나 글이 업로드가 되지 않는다. 리스트뷰에 해당하는 부분도 데이터를 읽어와서 최신순으로 나열해야 하지만, 데이터베이스에 저장된 것이 없기 때문에 이 모든 것을 확인하지 못해 아쉬웠다. 수업시간에 배웠던 내용과 실습 자료를 바탕으로 코드를 구현하였지만, 자바에 익숙하지 않아서 코드를 이해하는 것에도 큰 어려움이 있었다. 한 줄 한 줄 이해해보고, 수업시간에 배웠던 개념들을 바탕으로 안드로이드에 해당하는 코드는 구현할 수 있었던 것 같다. 또한 앱 바에 해당하는 부분(Proj_and_2021202058)과 ADD 버튼이 한 줄에 나타나는 것이 아니라 다음 줄에 나타나 이를 해결하려고 했으나, 코드로 구현한 적이 없어 기본으로 생기는 부분이라고 생각되어 고치지 않았다. 이번 기회를 통해 커리큘럼 이외의 언어도 더 공부해보고, 자바에 대해서도 더 공부하고 코드를 많이 짜봐야 할 것 같다는 생각이 많이 들었다.