

모바일프로그래밍

최종보고서

2016. 06. 13.

과목명	모바일프로그래밍
주 제	내가 꾸미는 우리 집
팀 명	오구오구
팀 장	20130903 김용훈
팀 원	20130891 김경희 20130895 김미선 20130911 김헌진
담당교수	최정열

목차

1. 프로젝트 제안서	1
가. 개요	1
1) 주제명	1
2) 목적	1
3) 배경	1
4) 필요성	1
5) 목표	2
나. 개발범위	2
1) 대상 업무	2
2) 개발환경	2
다. 개발 추진체계	3
1) 팀 소개	3
2) 개발 수행 추진체계	4
3) 추진 방법	4
라. 개발 추진 일정	4
마. 설계 제한 요소 반영	5
1) 경제성	5
2) 편리성	5
3) 윤리성	5
4) 안정성	5
5) 유지 관리 용이성	5
바. 기대효과	5
2. 요구분석서	6
가. 기존 서비스 분석	6
1) 아이디어스	6
2) 오늘의 집	7
3) DIY리폼	8
나. 요구사항 분석서	9
1) 기존 서비스 분석에 따른 요구 사항	9
2) 서비스 개발 요구사항 분석	11
다. 서비스 개발 항목	21
1) 서비스 개발 항목	21
2) 서비스 기능 흐름도	21
라. Use Case Diagram	22
1) 유즈케이스 다이어그램	22
2) 유즈케이스 명세서 - 사용자	23
3) 유즈케이스 명세서 - 판매자	33

4) 유즈케이스 명세서 - 관리자	35
5) 유즈케이스 명세서 - 재료 판매업체	38
6) 유즈케이스 명세서 - 배송업체	38
7) 유즈케이스 명세서 - 결제 시스템	39
 3. 시스템 설계서	40
가. 개요	40
1) 시스템의 목표	40
2) 시스템의 주요 기능	40
3) 설계상 제약 사항	40
나. 시스템 구조	40
1) 시스템 구조 개요	40
2) 시스템 구조도	41
다. 소프트웨어 설계	41
1) 클래스 다이어그램(class diagram)	41
2) 시퀀스 다이어그램(sequence diagram)	42
3) 활동 다이어그램(Activity Diagram)	42
라. 데이터베이스 설계	47
1) ER diagram	47
2) DB 테이블 기술서	48
마. 사용자 인터페이스	59
1) UI 설계서	59
 4. 코딩보고서	
가. 프로그램 명명규칙	65
1) 패키지 이름	65
2) 타입 이름	65
3) 메소드 이름	67
4) 변수 이름	67
5) 주석 규칙	68
나. 프로그램 목록	69
1) 전체 프로그램의 이름과 하는 일	69
2) 안드로이드 스튜디오 파일 구조	74
다. 프로그램 소스	77

1. 프로젝트 제안서

가. 개요

1) 주제명

- 인테리어 소품 및 재료 판매채널 확보와 노하우 공유를 위한 모바일 애플리케이션 제작

2) 목적

가) 오구오구 팀

- 소품 판매자와 소품 구매자 사이의 **중계자 역할**을 통해 수익 창출
- 소프트웨어 개발 방법론을 접목시켜 체계적인 설계 절차를 거친 개발 경험 축적

나) 소품 판매자

- 부족한 소품 판매 환경을 개선하기 위한 인테리어 소품의 **판매채널 확보**
- 경력단절여성, 시니어, 주부 등이 인테리어 소품을 판매하여 새로운 경제활동 가능
- 인테리어 소품 제작의 노하우 공유를 통해 인지도 증가

다) 소품 구매자

- 인테리어 소품 제작의 노하우에 사용된 재료들을 손쉽게 구매

라) 재료 판매업체

- 인테리어 소품 제작에 사용된 재료 판매를 통해 수익 창출 및 홍보 효과

3) 배경

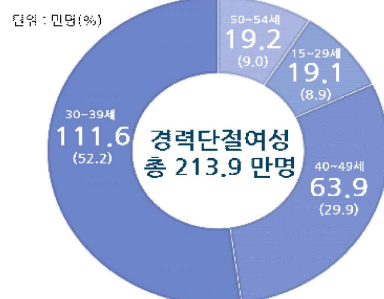
- 집을 스스로 꾸미는 **셀프인테리어 집방**에 대한 관심 증가
- 방송 및 각종 SNS를 통해 자신의 집안 인테리어를 공유하는 문화 확산
- 합리적인 비용으로 자신의 공간을 꾸미길 원하는 1인 가구, 시니어 계층 및 경력단절여성들에 의해 DIY(Do It Yourself)의 인기가 증가

4) 필요성

가) 경력단절여성의 경제활동 어려움

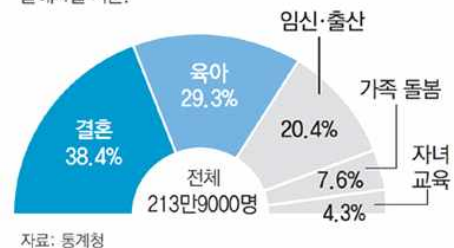
- 결혼, 육아, 임신·출산 등으로 일을 하기에 버거운 상황이므로 시간이나 장소에 제약받지 않는 **자유로운 경제활동**을 원함

2014년 연령대별 경력단절여성 규모 (자료:통계청)



[그림 1] 연령대별 경력단절여성 규모

경력단절여성 경력단절 이유
올해 4월 기준.



자료: 통계청

2014년 경력단절여성 214만명

[그림 2] 경력단절여성의 경력단절 이유

나) 한정된 거래처

- 현재 인테리어 소품 시장은 기업 위주
- 개인의 경우 판매채널이 매우 적음
- DIY를 즐겨하는 구매자는 여러 재료를 찾아 구매하기 번거로움

다) 결론

- 경력단절여성의 경제 활동에 도움을 줄 수 있는 시스템이 필요
- 소품 판매자와 소품 구매자를 연결해주는 판매채널이 필요
- 재료 판매업체와 협약을 통해 재료 구매의 간편화가 필요

5) 목표

- 소품 판매자가 자신만의 인테리어 소품 판매 및 노하우를 공유할 수 있는 모바일 애플리케이션 제작
- 소품 구매자가 인테리어 소품의 재료 및 완제품을 구매할 수 있도록 결제시스템 구축

나. 개발 범위

1) 대상 업무

가) H/W 구축 범위

- 해당사항 없음

나) S/W 개발 범위



[그림 3] S/W 개발 범위

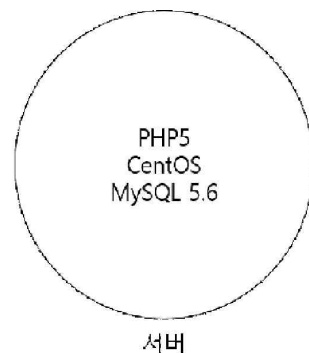
2) 개발환경



[그림 4] H/W 개발 환경



[그림 5] S/W 개발 환경



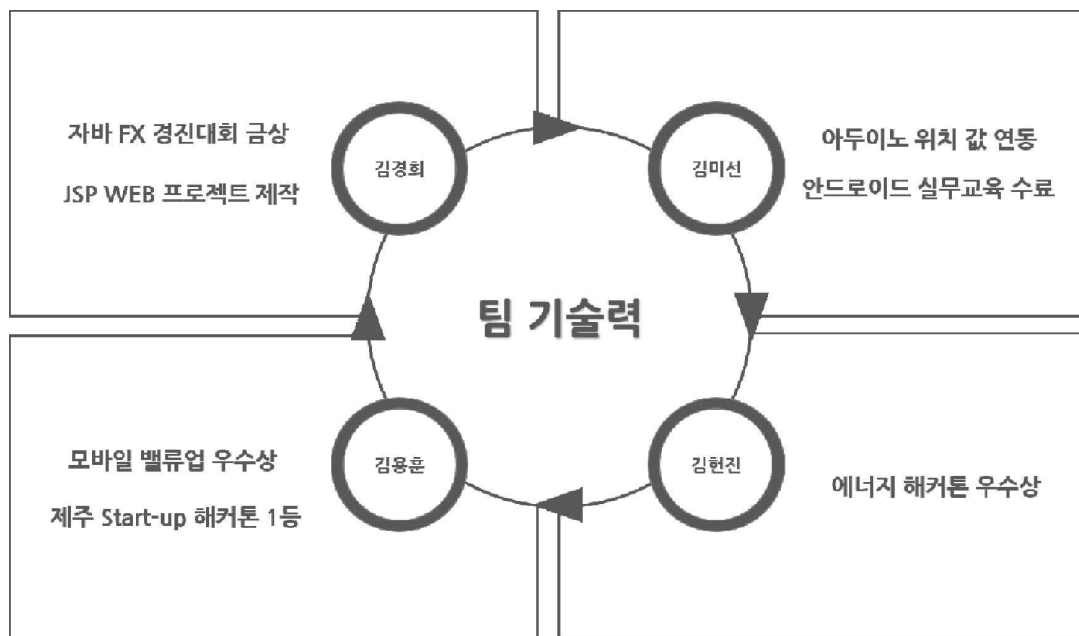
다. 개발 추진체계

1) 팀 소개

가) 팀 형성 배경

2015년 겨울 방학, 팀장이 실무 교육을 중도포기 없이 완수한 학우들을 찾아가 설득하여 팀 형성이 되었다. 팀원 모두 같은 학번으로 3년간 꾸준히 친밀감을 쌓아왔고 서로의 장점을 잘 알기 때문에 성공적으로 프로젝트를 마무리 할 수 있다.

나) 팀원 기술력



[그림 6] 팀원 기술력

다) 팀 성공요소

위에 작성한 바와 같이 팀원들이 증명 가능한 기술력을 보유하고 있기 때문에 프로젝트 수행에 있어 기술적으로 큰 어려움이 없다.

라) 취약점 분석

- 전문적인 디자이너가 없다.

마) 극복방안

- 색상조합에 대한 공부를 한다.
- 디자이너에게 피드백을 받는다.
- 디자인이 잘 된 모바일 애플리케이션을 자주 본다.

2) 개발 수행 추진체계



[그림 7] 개발 수행 추진체계

3) 추진 방법

- 현재 존재하는 여러 모바일 애플리케이션을 벤치마킹하여 차별성 및 개선점 등을 조사
- 한 달 단위로 프로토타입을 만들고 고객의 요구사항을 반영하여 개선
- 개발 완료와 프로젝트 마감일에 충분한 시간을 두고 많은 테스트를 거쳐 부족한 점을 보완
- 기술적으로 부족한 부분은 멘토님 혹은 교수님과의 면담을 통해 보완

라. 개발 추진 일정

일정	세부과제	개발내용
4월	요구사항 수집 및 분석 문서작성	- 고객의 요구사항을 수집하여 분석 - 제안서 및 중간보고서 작성
5월	개발 환경 구축 모바일 애플리케이션 완성	- 이미지를 저장하기 위한 서버 및 회원 정보를 저장하기 위한 DB를 구축 - 인테리어 소품 제작의 노하우 공유 기능이 있는 모바일 애플리케이션을 개발
6월	유지보수 모바일 애플리케이션 런칭	- 모바일 애플리케이션의 안정성을 위한 반복적인 유지보수 - 완성된 모바일 애플리케이션을 Play스토어에 등록
7월	유지보수 애플리케이션을 활용한 사업 진행	- 모바일 애플리케이션의 안정성을 위한 반복적인 유지보수 - 홍보를 통해 고객을 유치하고 사업을 진행
8월		
9월		
10월		
11월		

마. 설계 제한 요소 반영

1) 경제성

- 개방형 모바일 플랫폼 안드로이드를 사용하여 개발 비용 감소

2) 편리성

- UI/UX(User Experience)를 고려한 디자인

3) 윤리성

- 오픈소스나 다른 코드를 참고하여 사용하는 경우 출처와 원저자의 이름을 명시

4) 안정성

- 개발 방법론을 활용한 설계 절차를 거쳐 프로젝트를 진행
- Unit(단위) 테스트를 통해 안정성을 보장

5) 유지 관리 용이성

- 문서작업을 통해 유지관리 용이성 증대
- MVC패턴을 통해 각 부분을 세분화시켜 관리 및 재사용성을 높임

바. 기대효과

1) 오구오구 팀

- 소품 판매자와 소품 구매자 사이의 **중계자 역할**을 하여 발생하는 수수료로 수익 창출
- 재료 판매업체와 협약을 맺어 **재료 판매**로 인한 수수료로 수익 창출

2) 소품 판매자

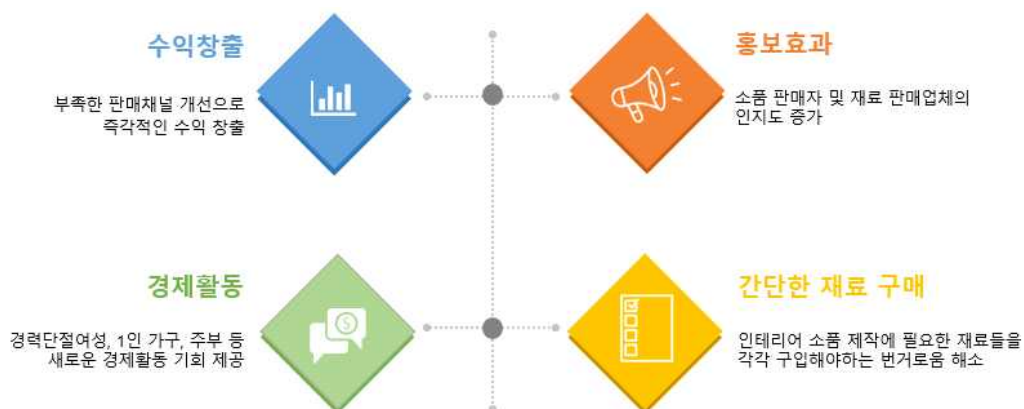
- 부족한 **판매채널 개선**으로 즉각적인 수익 창출
- 경력단절여성, 1인 가구, 시니어, 주부 등이 인테리어 소품 제작을 통해 새로운 경제활동 기회 제공

3) 소품 구매자

- 인테리어 소품 제작에 필요한 여러 재료들을 각각 구입해야하는 번거로움 해소
- 소품 판매자의 노하우를 응용하여 직접 자신만의 인테리어 소품을 제작

4) 재료 판매업체

- 인테리어 소품에 사용된 재료 판매를 통해 수익 창출



[그림 8] 기대효과

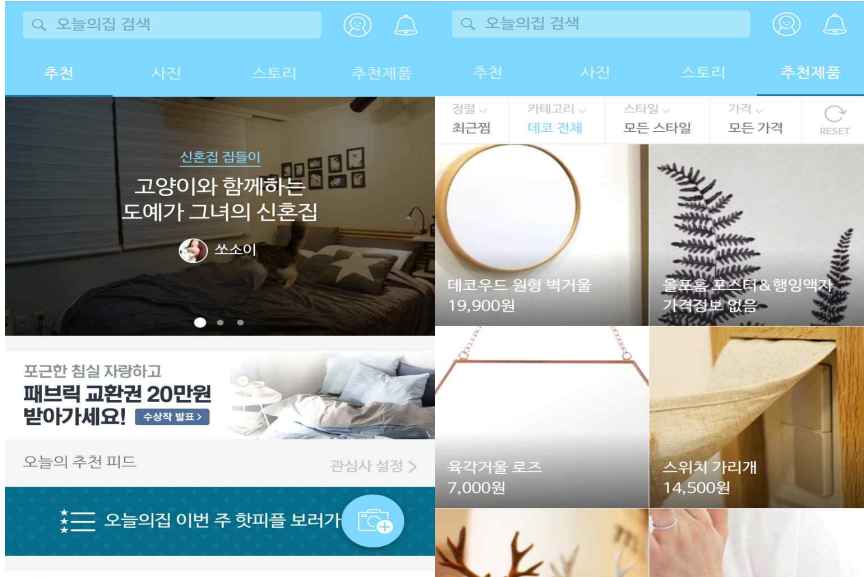
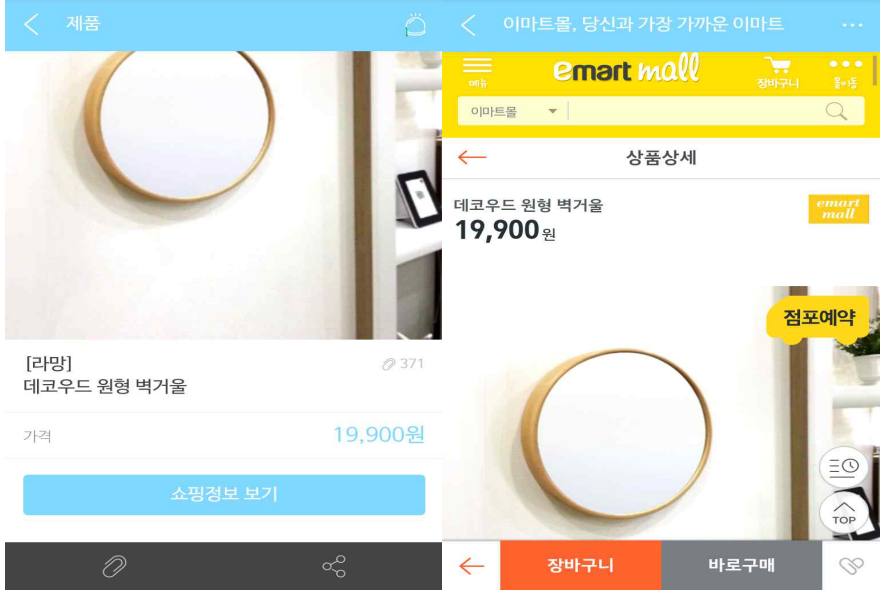
2. 요구 분석서

가. 기존 서비스 분석

1) 아이디어스

<p>화면</p>	 <p>[그림 9] '아이디어스' 인테리어 소품 판매 화면</p>  <p>[그림 10] 악세사리 디자이너와 운영 중인 홈페이지</p>
<p>분석</p>	<ul style="list-style-type: none"> - 인테리어 소품 제작의 노하우를 공유하지 않음 - 전문적인 디자이너의 인테리어 소품만을 판매 - 일반 사람이 인테리어 소품을 판매할 경우 제약이 존재 - 인테리어 소품 제작에 사용되는 재료를 판매하지 않음
<p>내꾸우만의 차별성</p>	<ul style="list-style-type: none"> - 인테리어 소품 제작의 노하우를 공유 - 전문적인 디자이너뿐만 아니라 일반 사람들도 인테리어 소품을 판매 - 인테리어 소품 제작에 사용되는 재료를 판매

2) 오늘의 집

<p>화면</p>	 <p>[그림 11] '오늘의 집' 인테리어 소품 판매 화면</p>  <p>[그림 12] 구매 시 판매 사이트로 연결되는 화면</p>
<p>기능분석</p>	<ul style="list-style-type: none"> - 인테리어 소품 제작의 노하우를 공유하지 않음 - 일반사람들이 인테리어 소품 판매 가능 - 인테리어 소품을 구매하기 위해 또 다른 사이트로 연결이 되는 불편함 존재 - 인테리어 소품 제작에 사용되는 재료를 판매하지 않음
<p>내꾸우만의 차별성</p>	<ul style="list-style-type: none"> - 인테리어 소품 제작의 노하우를 공유 - 모바일 애플리케이션 내에서 인테리어 소품을 구매 - 인테리어 소품 제작에 사용되는 재료를 판매

3) DIY리폼

<p>화면</p>	<div data-bbox="464 309 1319 882"> </div> <p>[그림 13] 'DIY리폼'의 로딩화면과 메인화면</p> <div data-bbox="459 947 1326 1534"> </div> <p>[그림 14] 인테리어 소품 제작의 노하우 공유 화면</p>
<p>기능분석</p>	<ul style="list-style-type: none"> - DIY 소품 제작의 노하우를 공유하고 있음 - 인테리어 소품을 판매하지 않음 - 인테리어 소품 제작에 사용되는 재료를 판매하지 않음 - UI/UX를 고려하지 않은 디자인
<p>내꾸우만의 차별성</p>	<ul style="list-style-type: none"> - 모바일 애플리케이션 내에서 인테리어 소품을 판매 - 인테리어 소품 제작에 사용되는 재료를 판매

나. 요구사항 분석서

1) 기존 서비스 분석에 따른 요구 사항

가) 인터뷰 대상

인터뷰 대상은 20대 여성으로, 현재 자신이 직접 인테리어 소품을 제작하고 블로그에 그 과정을 업로드하고 있다. 또한 유튜브에 제작 과정을 동영상으로 촬영하여 업로드하고 있다.

블로그 :

<http://blog.naver.com/niceoma>

유튜브 :

<https://www.youtube.com/channel/UC5OaQHImDBGJNG-h1tludBA>



[그림 15] 인터뷰 대상이 운영하는 블로그

나) 인터뷰 내용

(질문1) 직접 이쁘고 실용적인 소품들을 많이 제작하고 계시는데 제작 노하우를 블로그에 올리는 이유가 있으신가요?

처음에는 취미로 소품들을 만들고 노하우를 블로그에 올리기로 시작했어요. 그러다보니 많은 분들이 어떻게 만드냐고 물어보시고, 재료는 어디서 구매했냐고 많이 궁금해 하시더라고요. 그래서 좀 더 자세한 정보를 올려야겠다 싶어서, 제작 노하우 글들을 작성하기 시작했어요. 사진으로는 설명이 부족하고 만드는 팁들을 담기 어려워서, 보고 따라 만들기 쉽도록 영상을 시작하게 되었고요.

(질문2) 소품을 제작하는데 필요한 재료를 어디에서 구매하시나요? 재료를 구매하는 데 불편한 점은 없으신가요?

필요한 재료들을 구상을 한 다음에, 포털 사이트에서 하나씩 검색해서 찾아요. 검색하면 비슷한 것들이 나오기는 하는데, 그것을 각각 다른 사이트에서 구매하려니 배송비가 부담이 되더라고요. 한 곳에서 모두 구매할 수 있다면 참 좋을 텐데, 그게 참 아쉬워요. DIY 키트를 판매하는 곳은 편리하긴 한데, 가격이 비싼 편이고 원하는 대로 재료를 구성할 수가 없어서 잘 이용하지 않게 되더라고요.

(질문3) 인테리어 소품을 만드는데 필요한 재료를 하나의 사이트에서만 구매하시나요?

아니요. 여러 곳에서 찾아서 사야해요. 소품 만들기의 특성상 작은 재료들이 다양하게 필요한 편인데, 업체별로 취급하는 재료들이 다르거든요. 하나씩 구매를 해야 해서 보통 대량으로 구매하고 있어요. 그래서 한 번 살 때 재료비가 많이 드는 편이죠.

(질문4) 다른 블로그에서 공유하는 제작 노하우를 보고 직접 만들고 싶은 적이 있으신가요?

네 그럼요. 하마미, 제이썬, 특전사마누라 등 셀프인테리어 노하우를 공유하는 분들과 블로그 이웃을 맺고 꾸준히 노하우를 보고 있어요. 사진과 함께 글로 설명을 해 주시는데, 그런 글들을 읽고 응용해서

만들어 본 적도 있어요. 페인트칠 하나를 하려 해도 여러 가지 상식이 있어야 하는데, 책에서 배우는 것보다 실제 해본 분들의 노하우를 듣는 게 빠르고 배우기 쉽거든요.

(질문5) 제작한 물품을 판매할 의향이 있으신가요?

판매할 수 있다면 하고 싶어요. 만들어진 소품들이 있는데, 인테리어 소품이 많다고 좋은 건 아니거든요. 적재적소에 위치해야 아름답게 꾸밀 수 있는데, 필요한 분들이 있다면 판매하고 보람도 느끼고 그럴 수 있었으면 좋겠어요.

(질문6) 만약 제작한 물품을 판매하는 중이시라면 어떤 채널을 활용하시나요?(애플리케이션, 사이트 등)

(질문7) 판매할 의향이 있는데 물품을 판매하지 않는다면 어떤 이유 때문인가요?

(6,7번 답변) 사실 판매할 수 있는 사이트나 애플리케이션이 없어서 판매채널 확보가 가장 어려워요. 아이디어스에 작가로 등록하기에는 업체위주의 제품이고, 그렇다고 중고나라 같은 일반인 거래 사이트도 맞지 않고요. 그래서 요즘엔 플리마켓에 나가는 것을 계획하고 있어요. 한 달에 한 번 정도 지역별로 플리마켓을 하는데 직접 판매를 할 수 있거든요. 하지만 플리마켓에서 자리를 얻는 것도 신청 후 선정이 되어야 해서 어려움이 많아요.

(질문8) 수익이 발생한다면 어느 경로로 발생하나요?(광고, 판매 등)

광고수익이에요. 네이버 애드포스트나 유튜브 광고수익이 조금씩 발생하고 있어요. CPC 광고라서 마케팅을 열심히 해야 해요.

(질문9) 만약 노하우를 올리고 상품을 판매할 수 있는 애플리케이션이 있다면 사용하실 의향이 있으신가요?

네 그럼요! 노하우를 공유하면서 수익을 얻을 수 있는 채널이 생긴다면 알릴 수 있는 마케팅의 효과까지 일석삼조겠네요. DIY 관련 블로거분들에게 정말 기쁜 소식 일거예요.

다) 인터뷰 결론

인터뷰 대상자는 인테리어 소품 제작 노하우를 블로그에 업로드한다. 인터뷰 대상자의 블로그를 방문하여 제작 방법을 참고해 인테리어 소품을 만드는 이용자들이 있다. 이런 블로그 방문자들은 종종 블로거에게 소품 제작에 필요한 재료를 묻는다. 또한, 인터뷰 대상자도 다른 블로그를 방문해 소품 제작 방법을 배우고 있다.

현재 인터뷰 대상자는 만든 인테리어 소품을 판매하길 원한다. 하지만, 판매할 채널이 부족하다. 또한, 여러 재료를 각각 구입해야 하는 번거로움이 있다.

따라서, 인테리어 소품 제작 노하우를 공유하고 제품을 판매하길 원하는 소상공인들을 위해 글을 등록하고 만든 소품을 판매할 수 있는 애플리케이션이 필요하다. 또한, 노하우를 보고 완제품을 구입하거나 손쉽게 재료를 구입하길 원하는 사용자들을 위한 애플리케이션이 필요하다.

2) 서비스 개발 요구사항 분석

□ 기본사항

프로젝트명	내가 꾸미는 우리 집(내꾸우)		
작성자	김미선	작성일	2016.04.22

□ 용어

사용자	U: User
판매자	S: Seller
관리자	M: Manager
재료 판매업체	IV: Ingredient Vendor
배송업체	DE: Delivery Enterprise
결제 시스템	PS: Pay System

□ 요구사항

구분	요구사항	ID	비고
사용자	로그인	U-001	
	로그아웃	U-002	
	내 정보 관리	U-003	
	판매자 등록	U-004	
	회원탈퇴	U-005	
	노하우 조회	U-006	
	노하우 스크랩	U-007	
	재료 조회	U-008	
	노하우 및 재료 검색	U-009	
	소품 구매	U-010	
	재료 구매	U-011	
	구매한 내역 및 스크랩 내역 조회	U-012	
	배송위치 확인	U-013	
	수령확인 전송	U-014	
	커뮤니티 글 작성	U-015	
	커뮤니티 글 조회	U-016	
	댓글 작성	U-017	
	공지사항 조회	U-018	
	FAQ 조회	U-019	
판매자	노하우 작성	S-001	
	노하우 관리	S-002	
	판매한 내역 조회	S-003	
	소품 발송	S-004	
관리자 (오구오구)	판매자 등록 관리	M-001	
	공지사항 관리	M-002	
	FAQ 관리	M-003	
	거래 관리	M-004	
	푸쉬 알림	M-005	
	판매금액 송금	M-006	
재료 판매업체	재료 발송	IV-001	
배송업체	위치정보 제공	DE-001	
결제 시스템	결제	PS-001	

구 분		사용자		
요구사항ID		U-001	요구사항 명	로그인
개 요		로그인 기능		
요구 사항 내역	상세설명	사용자는 카카오톡 로그인을 통해 로그인 할 수 있다.		
	중요도	상	난이도	상

구 분		사용자		
요구사항ID		U-002	요구사항 명	로그아웃
개 요		로그아웃 기능		
요구 사항 내역	상세설명	사용자는 로그아웃 할 수 있다.		
	중요도	중	난이도	중

구 분		사용자		
요구사항 ID		U-003	요구사항 명	내 정보 관리
개 요		개인정보 조회/수정 기능		
요구 사항 내역	상세설명	사용자는 자신의 개인정보(사용자 이미지, 닉네임, 주소, 계좌명의, 은행명, 계좌번호, 휴대폰번호)를 조회하거나 수정할 수 있다.		
	중요도	상	난이도	중

구 분		사용자		
요구사항 ID		U-004	요구사항 명	판매자 등록
개 요		판매자 등록 신청 기능		
요구 사항 내역	상세설명	<p>사용자가 계좌명의, 은행명, 계좌번호를 입력하지 않은 경우 사용자 정보 수정 화면으로 가서 작성할 수 있다.</p> <p>사용자가 계좌명의, 은행명, 계좌번호를 입력한 경우 판매자 등록을 신청할 수 있다.</p>		
	중요도	상	난이도	중

구 분		사용자		
요구사항 ID		U-005	요구사항 명	회원탈퇴
개 요		회원탈퇴 기능		
요구 사항 내역	상세설명	<p>사용자는 회원탈퇴를 할 수 있다.</p> <p>회원탈퇴 시 사용자의 개인정보는 삭제되며 로그인 페이지로 간다.</p>		
	중요도	하	난이도	하

구 분		사용자		
요구사항 ID		U-006	요구사항 명	노하우 조회
개 요		인테리어 소품제작의 노하우 조회 기능		
요구 사항 내역	상세설명	<p>사용자는 인테리어 소품제작의 노하우를 조회할 수 있다.</p> <p>노하우 조회 시 작성자, 제작 정보(난이도, 소요시간, 소요비용), 제작 순서(사진, 설명) 등을 볼 수 있다.</p>		
	중요도	상	난이도	중

구 분		사용자		
요구사항 ID		U-007	요구사항 명	노하우 스크랩
개 요		노하우 스크랩 기능		
요구 사항 내역	상세설명	<p>사용자는 마음에 드는 인테리어 소품 제작의 노하우를 스크랩할 수 있다.</p> <p>스크랩한 노하우는 내 정보 화면에서 조회할 수 있다.</p>		
	중요도	중	난이도	중

구 분		사용자		
요구사항 ID		U-008	요구사항 명	재료 조회
개 요		인테리어 소품제작에 필요한 재료 조회 기능		
요구 사항 내역	상세설명	<p>사용자는 인테리어 소품 제작에 필요한 재료를 조회할 수 있다.</p> <p>재료 조회 시 재료 사진과 설명, 가격, 판매처 정보를 볼 수 있다.</p>		
	중요도	상	난이도	중

구 분		사용자		
요구사항 ID		U-009	요구사항 명	노하우 및 재료 검색
개 요		인테리어 소품제작의 노하우와 재료 검색 기능		
요구 사항 내역	상세설명	<p>사용자는 인테리어 소품 제작의 노하우와 소품 제작에 필요한 재료를 검색할 수 있다.</p> <p>사용자는 노하우의 내용이나 제목 또는 재료의 이름을 통해 검색이 가능하다.</p>		
	중요도	중	난이도	상

구 분		사용자		
요구사항 ID		U-010	요구사항 명	소품 구매
개 요		인테리어 소품 구매 기능		
요구 사항 내역	상세설명	사용자는 인테리어 소품을 구매할 수 있다.		
	중요도	상	난이도	상

구 분		사용자		
요구사항 ID		U-011	요구사항 명	재료 구매
개 요		인테리어 소품 제작 재료 구매 기능		
요구 사항 내역	상세설명	사용자는 인테리어 소품 제작에 필요한 재료를 구매할 수 있다.		
	중요도	상	난이도	상

구 분		사용자		
요구사항 ID		U-012	요구사항 명	구매한 내역 및 스크랩 내역 조회
개 요		구매한 내역과 스크랩 내역 조회 기능		
요구 사항 내역	상세설명	사용자는 자신이 구매한 인테리어 소품이나 재료의 내역을 조회할 수 있다. 또한 자신이 스크랩한 내역을 조회할 수 있다.		
	중요도	중	난이도	하

구 분		사용자		
요구사항 ID		U-013	요구사항 명	배송위치 확인
개 요		구매한 소품 및 재료의 배송 조회 기능		
요구 사항 내역	상세설명	사용자는 자신이 구매한 인테리어 소품이나 소품 제작에 필요한 재료에 대한 배송위치를 확인할 수 있다.		
	중요도	하	난이도	중

구 분		사용자		
요구사항 ID		U-014	요구사항 명	수령확인 전송
개 요		관리자에게 수령확인 메시지 전송 기능		
요구 사항 내역	상세설명	사용자는 인테리어 소품이나 재료를 배송 받으면 수령 받았다는 메시지를 관리자에게 전송할 수 있다.		
	중요도	중	난이도	중

구 분		사용자		
요구사항 ID		U-015	요구사항 명	커뮤니티 글 작성
개 요		커뮤니티 글 작성 기능		
요구 사항 내역	상세설명	사용자는 다른 사용자와 소통할 수 있는 커뮤니티에 글을 작성할 수 있다.		
	중요도	하	난이도	중

구 분		사용자		
요구사항 ID		U-016	요구사항 명	커뮤니티 글 조회
개 요		커뮤니티 글 조회 기능		
요구 사항 내역	상세설명	<p>사용자는 다른 사용자와 소통할 수 있는 커뮤니티에 글을 조회할 수 있다.</p> <p>커뮤니티 조회 시 작성자와 커뮤니티 글 내용, 댓글 수 와 다른 사용자가 남긴 댓글을 확인할 수 있다.</p> <p>조회한 글이 자신이 작성한 커뮤니티 글이라면 수정/삭제가 가능하다.</p>		
	중요도	하	난이도	중

구 분		사용자		
요구사항 ID		U-017	요구사항 명	댓글 작성
개 요		노하우 댓글 작성 기능		
요구 사항 내역	상세설명	<p>사용자는 조회하거나 검색해서 나온 인테리어 소품 제작의 노하우에 댓글을 작성할 수 있다.</p> <p>사용자는 다른 사용자나 자신이 작성한 커뮤니티 글에 댓글을 작성할 수 있다.</p>		
	중요도	하	난이도	중

구 분		사용자		
요구사항 ID		U-018	요구사항 명	공지사항 조회
개 요		공지사항 조회 기능		
요구 사항 내역	상세설명	사용자는 관리자가 작성한 공지사항을 조회할 수 있다.		
	중요도	하	난이도	하

구 분		사용자		
요구사항 ID		U-019	요구사항 명	FAQ 조회
개 요		FAQ 조회 기능		
요구 사항 내역	상세설명	사용자는 관리자가 작성한 자주 묻는 질문들을 조회할 수 있다.		
	중요도	하	난이도	하

구 분		판매자		
요구사항 ID		S-001	요구사항 명	노하우 작성
개 요		노하우 작성 기능		
요구 사항 내역	상세설명	<p>판매자는 사진이나 동영상 또는 글을 통해 인테리어 소품 제작의 노하우를 작성할 수 있다.</p> <p>판매자는 제목, 소요비용, 유튜브코드를 작성하고 카테고리, 난이도, 소요시간, 사진/글 등록, 소품판매를 선택할 수 있다.</p> <p>판매자는 자신이 제작한 소품을 판매하거나 사용된 재료를 등록할 수 있다.</p>		
	중요도	상	난이도	상

구 분		판매자		
요구사항 ID		S-002	요구사항 명	노하우 관리
개 요		노하우 관리 기능		
요구 사항 내역	상세설명	판매자는 자신이 작성한 노하우 글을 수정하거나 삭제할 수 있다.		
	중요도	중	난이도	중

구 분		판매자		
요구사항 ID		S-003	요구사항 명	판매한 내역 조회
개 요		판매한 내역 조회		
요구 사항 내역	상세설명	판매자는 자신이 판매한 상품의 내역을 조회할 수 있다.		
	중요도	중	난이도	중

구 분		판매자		
요구사항 ID		S-004	요구사항 명	상품 발송
개 요		인테리어 상품 발송 기능		
요구 사항 내역	상세설명	판매자는 관리자로부터 푸쉬 알림(구매자의 구매요청)을 받으면 인테리어 상품을 발송할 수 있다.		
	중요도	상	난이도	중

구 분		관리자(오구오구)		
요구사항 ID		M-001	요구사항 명	판매자 등록 관리
개 요		판매자 등록 관리 기능		
요구 사항 내역	상세설명	관리자는 사용자가 판매자 등록을 신청할 경우 사용자가 작성한 계좌 명과의 계좌번호의 명의를 일치할 경우에만 판매자로 등록한다.		
	중요도	상	난이도	중

구 분		관리자(오구오구)		
요구사항 ID		M-002	요구사항 명	공지사항 관리
개 요		공지사항 작성 기능		
요구 사항 내역	상세설명	관리자는 사용자들이나 판매자, 재로업체에게 공지할 사항을 작성할 수 있다. 또한 공지사항을 수정하거나 삭제할 수 있다.		
	중요도	상	난이도	하

구 분		관리자(오구오구)		
요구사항 ID		M-003	요구사항 명	FAQ 관리
개 요		FAQ 관리 기능		
요구 사항 내역	상세설명	관리자는 사용자, 판매자, 재료 판매업체가 자주 묻는 질문들을 작성할 수 있다. 또한 FAQ를 수정하거나 삭제할 수 있다.		
	중요도	하	난이도	하

구 분		관리자(오구오구)		
요구사항 ID		M-004	요구사항 명	거래 관리
개 요		거래 관리 기능		
요구 사항 내역	상세설명	관리자는 구매자로부터 수령 확인 메시지를 받아 구매자에게 인테리어 소품이 배송되었는지 확인을 할 수 있다. 구매자가 수령확인 메시지를 10일 이상 보내지 않을 경우 관리자는 소품이나 재료가 구매자에게 배송이 되었다고 간주한다.		
	중요도	상	난이도	중

구 분		관리자(오구오구)		
요구사항 ID		M-005	요구사항 명	푸쉬 알림
개 요		푸쉬 알림 기능		
요구 사항 내역	상세설명	사용자가 완제품을 결제 할 경우 자동으로 관리자에게 구매를 요청한다는 푸쉬 알림을 보낸다. 관리자가 푸쉬 알림을 받으면 판매자에게 구매 요청 알림을 보낸다. 판매자나 재료 판매업체가 배송정보를 입력하면 관리자에게 푸쉬 알림을 보낸다. 관리자는 판매자 또는 재료 판매업체가 입력한 배송정보를 확인하여 구매자에게 푸쉬 알림을 통해 알려줄 수 있다.		
	중요도	상	난이도	중

구 분		관리자(오구오구)		
요구사항 ID		M-006	요구사항 명	판매금액 송금
개 요		판매자에게 판매금액 송금 기능		
요구 사항 내역	상세설명	인테리어 소품 및 재료 배송이 완료된 경우 판매자에게 수수료를 제외한 나머지 판매금액을 송금할 수 있다.		
	중요도	상	난이도	하

구 분		재료 판매업체		
요구사항 ID		IV-001	요구사항 명	재료 발송
개 요		인테리어 소품 재료 배송 기능		
요구 사항 내역	상세설명	<p>사용자가 재료를 결제 할 경우 자동으로 재료 판매업체에게 구매를 요청한다는 푸쉬 알림을 받는다.</p> <p>재료 판매업체는 푸쉬 알림을 받아 구매자의 구매 요청을 확인하여 인테리어 소품 재료를 발송할 수 있다.</p> <p>재료 발송한 후 재료 판매업체는 배송정보를 입력하여 관리자에게 푸쉬 알림을 보낸다.</p>		
	중요도	상	난이도	중

구 분		배송업체		
요구사항 ID		DE-001	요구사항 명	위치정보 제공
개 요		위치정보 제공 기능		
요구 사항 내역	상세설명	사용자가 주문한 소품이나 재료에 대한 배송조회를 원하는 경우 배송업체는 현재 택배의 위치 정보를 제공해준다.		
	중요도	중	난이도	하

구 분		결제 시스템		
요구사항 ID		PS-001	요구사항 명	결제
개 요		결제 기능		
요구 사항 내역	상세설명	<p>사용자가 주문한 상품에 대해 결제한다.</p> <p>사용자가 인테리어 소품이나 재료를 구매하는 경우 결제 방식(무통장, 카드결제 등)을 선택한 후 결제 시스템이 결제를 승인한다.</p>		
	중요도	상	난이도	상

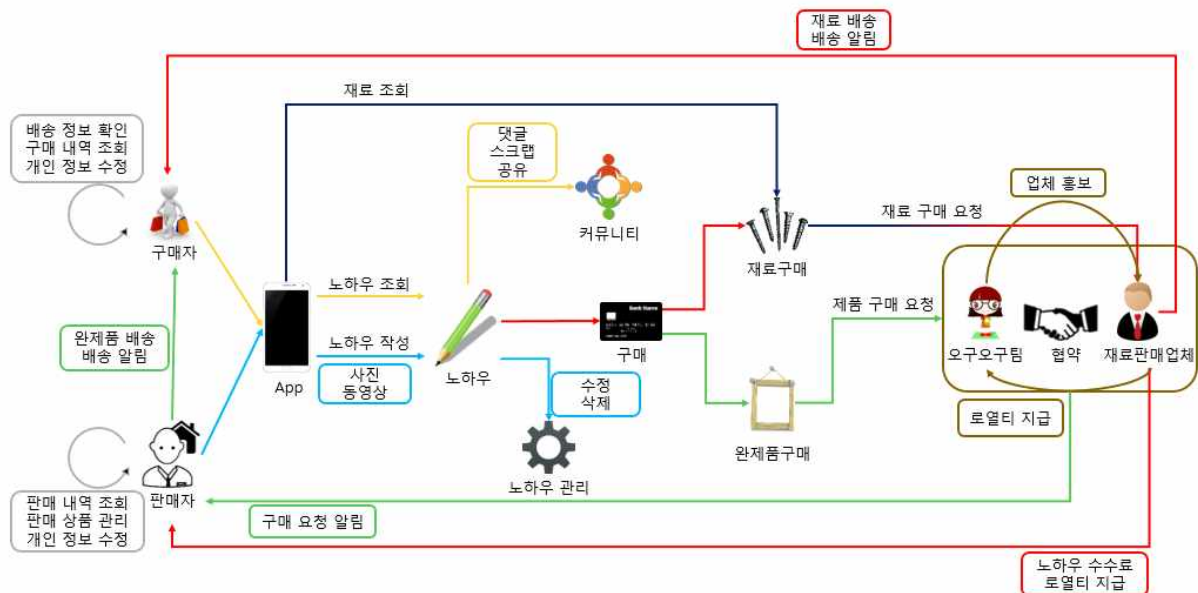
다. 서비스 개발 항목

1) 서비스 개발 항목

가) 모바일 애플리케이션 개발

- 인테리어 소품 판매 및 구매를 위한 결제 기능
- 노하우 공유 및 글 작성 기능
- 거래 정보 관리, 배송 조회 및 부가 기능
- 커뮤니티 기능

2) 서비스 기능 흐름도



[그림 16] 서비스 기능 흐름도

구매자 재료 구매 흐름도1

1. 구매자가 노하우를 보고 재료 구매
2. 재료판매업체가 구매자에게 재료 배송
3. 배송 시 구매자에게 알림
4. 재료판매업체는 노하우 작성자와 오구오구팀에게 수수료 제공

구매자 재료 구매 흐름도2

1. 구매자가 필요한 재료를 검색하여 조회
2. 재료 구매 흐름도1(1-2)로 이동

재료 판매업체와의 협약

1. 오구오구팀과 재료 판매업체 간 협약을 맺음
2. 애플리케이션을 통하여 재료 판매업체 홍보
3. 오구오구팀은 재료가 판매될 시 수수료 획득

커뮤니티 형성 흐름도

1. 사용자(구매자, 판매자)가 노하우 조회
2. 마음에 드는 노하우 스크랩 가능
3. 마음에 드는 노하우 공유 가능
4. 노하우에 댓글을 작성하여 커뮤니티 형성

구매자 완제품 구매 흐름도

1. 구매자가 노하우를 보고 완제품 구매
2. 오구오구팀이 소품 판매자에게 구매 요청 알림
3. 판매자가 구매자에게 완제품 배송
4. 배송 시 구매자에게 알림
5. 오구오구팀은 판매자와 구매자 사이에서 수수료 획득

소품 판매자 노하우 작성 흐름도

1. 판매자는 인테리어 소품 제작의 노하우를 작성
2. 노하우 작성에는 사진 및 동영상 업로드 가능
3. 작성자는 노하우 수정 및 삭제 가능

사용자 부가 기능

1. 구매자는 구매 내역, 배송 정보 조회 가능
2. 판매자는 판매 상품 내역 조회, 판매 상품 관리 가능
3. 사용자(구매자, 판매자)는 개인 정보 수정 가능

[그림 17] 서비스 기능 흐름도의 상세 설명

라. Use Case Diagram

1) 유즈케이스 다이어그램

유즈케이스 다이어그램	프로젝트명	작성일	작성자
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김용훈

1. 시스템 상황 분석

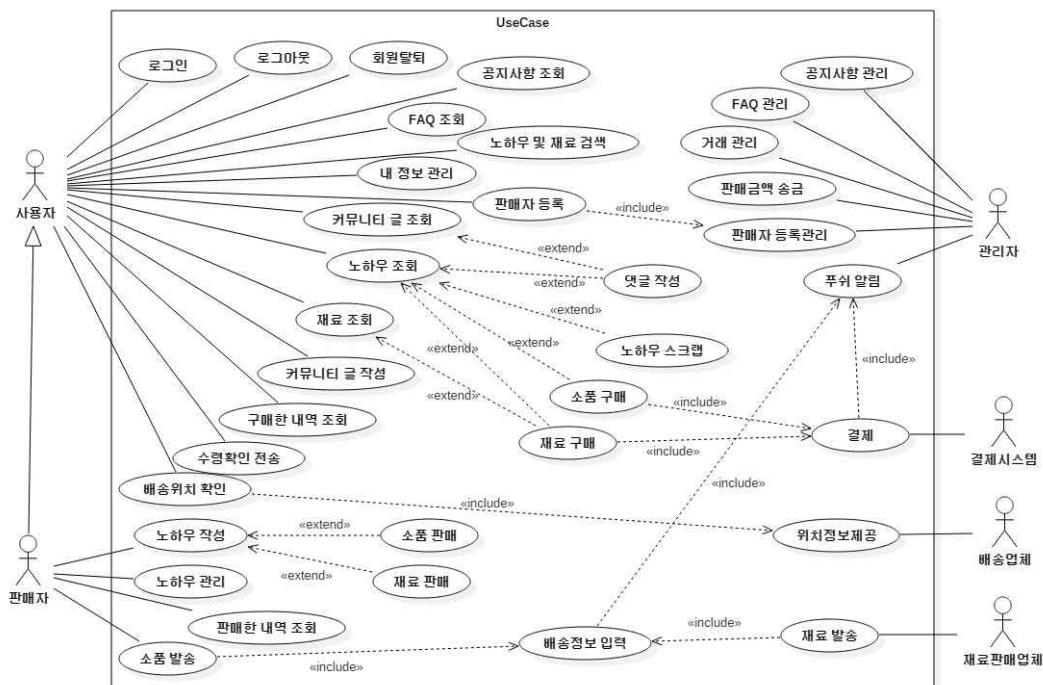
- 인테리어 소품 판매 및 구매를 위한 결제 기능
- 노하우 공유 및 글 작성 기능
- 거래 정보 관리, 배송 조회 및 부가 기능
- 커뮤니티 기능

2. 액터 식별



[그림 18] 액터 식별

3. 유즈케이스 다이어그램



[그림 19] 유즈케이스 다이어그램

2) 유즈케이스 명세서 - 사용자

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김용훈	U-001
<p>▶ 유즈케이스명 : 로그인</p> <p>▶ 액터명 : 사용자</p> <p>▶ 유즈케이스 개요 및 설명 : 사용자는 로그인을 할 수 있다.</p> <p>▶ 사전 조건 : 사용자는 카카오톡 회원가입을 한 상태이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 로그인 버튼을 누른다.</p>				

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김용훈	U-002
<p>▶ 유즈케이스명 : 로그아웃</p> <p>▶ 액터명 : 사용자</p> <p>▶ 유즈케이스 개요 및 설명 : 사용자는 로그아웃을 할 수 있다.</p> <p>▶ 사전 조건 : 사용자는 로그인한 상태이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 설정 버튼을 눌러 설정 페이지로 들어간다.</p> <p>(2) 로그아웃 버튼을 누른다.</p>				

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김용훈	U-003

▶ 유즈케이스명 : 내 정보 관리

▶ 액터명 : 사용자

▶ 유즈케이스 개요 및 설명 : 사용자는 내 정보를 조회, 수정할 수 있다.

▶ 사전 조건 : 사용자는 로그인한 상태이다.

□ 이벤트 흐름

- 정상 흐름
 - (1) 내 정보 버튼을 눌러 내 정보 페이지로 들어간다.
 - (2) 사용자 이미지, 내가 쓴 노하우, 스크랩한 노하우를 볼 수 있다. (A1)
- 대치흐름
 - (A1) 내 정보를 수정하길 원하는 경우 정보수정 버튼을 눌러 사용자 이미지, 닉네임, 주소, 계좌명의, 계좌번호, 휴대폰번호를 수정할 수 있다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김용훈	U-004

▶ 유즈케이스명 : 판매자 등록

▶ 액터명 : 사용자

▶ 유즈케이스 개요 및 설명 : 사용자는 판매자 등록을 할 수 있다.

▶ 사전 조건 : 사용자는 로그인한 상태이다. 사용자는 판매자가 아닌 상태이다.

□ 이벤트 흐름

- 정상 흐름
 - (1) 판매자 등록 버튼을 누른다. (A1)
 - (2) 신청 버튼을 누른다. (A2)
- 대치흐름
 - (A1) 사용자가 계좌명의, 은행명, 계좌번호를 입력하지 않은 경우 사용자 정보 수정 화면으로 이동시킨다.
 - (A2) 사용자가 작성한 계좌명의와 계좌번호의 명의가 일치하지 않은 경우. 반려한다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김용훈	U-005
<p>▶ 유즈케이스명 : 회원탈퇴</p> <p>▶ 액터명 : 사용자</p> <p>▶ 유즈케이스 개요 및 설명 : 사용자는 회원탈퇴를 할 수 있다.</p> <p>▶ 사전 조건 : 사용자는 로그인한 상태이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 설정 버튼을 눌러 설정 페이지로 들어간다.</p> <p>(2) 회원탈퇴 버튼을 누른다.</p> <p>(3) 로그인 화면으로 이동한다.</p>				

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김경희	U-006
<p>▶ 유즈케이스명 : 노하우 조회</p> <p>▶ 액터명 : 사용자</p> <p>▶ 유즈케이스 개요 및 설명 : 사용자가 인테리어 소품의 제작 노하우 글을 조회한다.</p> <p>▶ 사전 조건 : 승인된 사용자이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 사용자가 로그인을 한다.</p> <p>(2) 노하우 리스트를 확인한다.</p> <p>(3) 노하우 글을 클릭하여 작성자, 제작 정보(난이도, 소요시간, 소요비용), 제작 순서(사진, 설명)을 확인한다.</p>				

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김경희	U-007
<p>▶ 유즈케이스명 : 노하우 스크랩</p> <p>▶ 액터명 : 사용자</p> <p>▶ 유즈케이스 개요 및 설명 : 사용자가 마음에 드는 제품의 제작 노하우를 스크랩한다.</p> <p>▶ 사전 조건 : 사용자가 마음에 드는 제품이 있는 상태이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 사용자가 마음에 드는 노하우 글을 조회한다.</p> <p>(2) 글을 스크랩한다.</p>				

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김경희	U-008
<p>▶ 유즈케이스명 : 재료 조회</p> <p>▶ 액터명 : 사용자</p> <p>▶ 유즈케이스 개요 및 설명 : 사용자가 인테리어 소품의 제작에 필요한 재료를 조회한다.</p> <p>▶ 사전 조건 : 승인된 사용자이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 사용자가 로그인을 한다.</p> <p>(2) 재료 리스트를 확인한다.</p> <p>(3) 재료 글을 클릭하여 재료 사진과 설명, 가격, 판매처 정보를 확인한다.</p>				

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김용훈	U-009

▶ 유즈케이스명 : 노하우 및 재료 검색

▶ 액터명 : 사용자

▶ 유즈케이스 개요 및 설명 : 사용자는 노하우 및 재료 검색을 할 수 있다.

▶ 사전 조건 : 사용자는 로그인한 상태이다.

□ 이벤트 흐름

- 정상 흐름
 - (1) 검색 버튼을 눌러 검색 페이지로 들어간다.
 - (2) 키워드를 입력한다.
 - (3) 검색 버튼을 누른다. (A1)
 - (4) 노하우, 재료 글이 각각 리스트 형태로 나타난다.
- 대치흐름
 - (A1) 검색한 키워드에 대한 결과가 하나도 없을 경우, 다른 키워드로 검색하라는 메시지를 표시한다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김경희	U-010

▶ 유즈케이스명 : 소품 구매

▶ 액터명 : 사용자

▶ 유즈케이스 개요 및 설명 : 사용자가 마음에 드는 노하우 글을 보고 완성된 제품을 구매한다.

▶ 사전 조건 : 마음에 드는 노하우 글의 완제품의 재고가 있는 상태이다.

□ 이벤트 흐름

- 정상 흐름
 - (1) 사용자가 마음에 드는 노하우 글을 조회한다.
 - (2) 완제품의 수량을 선택한다. (A1)
 - (3) 구매하기 버튼을 눌러 결제한다.
- 대치흐름
 - (A1) 완제품의 재고가 없는 경우 표시 해준다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김경희	U-011

▶ 유즈케이스명 : 재료 구매

▶ 액터명 : 사용자

▶ 유즈케이스 개요 및 설명 : 사용자가 마음에 드는 인테리어 제품을 직접 만들기 위해 인테리어 소품 제작에 필요한 재료를 구매한다.

▶ 사전 조건 : 사용자가 노하우를 조회한 상태이다.

□ 이벤트 흐름

- 정상 흐름
 - (1) 사용자가 마음에 드는 노하우 글을 조회한다.
 - (2) 재료의 수량을 선택한다. (A1)
 - (3) 구매하기 버튼을 눌러 결제한다.
- 대치흐름
 - (A1) 재료의 재고가 없는 경우 표시 해준다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김경희	U-012

▶ 유즈케이스명 : 구매한 내역 및 스크랩 내역 조회

▶ 액터명 : 사용자

▶ 유즈케이스 개요 및 설명 : 사용자가 구매한 소품이나 재료에 대한 내역을 확인한다. 또한 스크랩한 내역을 조회한다.

▶ 사전 조건 : 사용자가 소품이나 재료를 구매한 상태이며 노하우를 스크랩한 상태이다.

□ 이벤트 흐름

- 정상 흐름
 - (1) 구매내역 버튼을 눌러 구매내역 페이지로 들어간다.
 - (2) 구매한 내역을 조회한다. (A1)
 - (3) 내 정보 버튼을 눌러 내 정보 페이지로 들어간다.
 - (2) 자신이 스크랩한 내역을 조회한다. (A2)
- 대치흐름
 - (A1) 구매한 내역이 없는 경우 표시 해준다.
 - (A2) 스크랩한 내역이 없는 경우 아무것도 표시하지 않는다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김경희	U-013

▶ 유즈케이스명 : 배송위치 확인

▶ 액터명 : 구매자

▶ 유즈케이스 개요 및 설명 : 구매자가 주문한 제품에 대한 배송위치를 조회한다.

▶ 사전 조건 : 사용자가 상품을 주문한 상태이다.

□ 이벤트 흐름

- 정상 흐름
 - (1) 사용자가 상품을 주문한다.
 - (2) 주문한 상품의 배송위치를 확인한다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김경희	U-014

▶ 유즈케이스명 : 수령확인 전송

▶ 액터명 : 구매자

▶ 유즈케이스 개요 및 설명 : 사용자가 주문한 상품을 수령했다는 사실을 관리자에게 전송한다.

▶ 사전 조건 : 사용자가 주문한 상품을 수령한 상태이다.

□ 이벤트 흐름

- 정상 흐름
 - (1) 사용자가 상품을 주문한다.
 - (2) 판매자나 재료업체에서 제품을 택배를 통해 보낸다.
 - (3) 사용자가 수령확인을 전송한다. (A1)
- 대치흐름
 - (A1) 상품을 수령하지 못한 경우 관리자에게 알린다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김경희	U-015

▶ 유즈케이스명 : 커뮤니티 글 작성

▶ 액터명 : 구매자

▶ 유즈케이스 개요 및 설명 : 사용자는 커뮤니티 글 작성을 할 수 있다.

▶ 사전 조건 : 사용자는 로그인한 상태이다.

□ 이벤트 흐름

- 정상 흐름
 - (1) 커뮤니티 탭을 눌러 커뮤니티 페이지로 들어간다.
 - (2) 글 작성 버튼을 눌러 글 작성 페이지로 들어간다.
 - (3) 글 내용을 입력한다.
 - (4) 완료 버튼을 누른다. (A1)
- 대치흐름
 - (A1) 글 내용을 입력하지 않은 경우, 글 내용을 입력하지 않음을 표시하고 제목 입력창으로 커서를 이동시킨다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김용훈	U-016

▶ 유즈케이스명 : 커뮤니티 글 조회

▶ 액터명 : 사용자

▶ 유즈케이스 개요 및 설명 : 사용자는 커뮤니티 글 조회를 할 수 있다.

▶ 사전 조건 : 사용자는 로그인한 상태이다.

□ 이벤트 흐름

- 정상 흐름
 - (1) 커뮤니티 탭을 눌러 커뮤니티 페이지로 들어간다.
 - (2) 커뮤니티에 등록된 글들을 리스트 형태로 나타낸다.
 - (3) 커뮤니티 글을 선택하여 작성자와 글 내용, 댓글을 확인한다. (A1)
- 대치흐름
 - (A1) 자신이 작성한 커뮤니티 글인 경우, 수정이 가능하다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김용훈	U-017
<p>▶ 유즈케이스명 : 댓글작성</p> <p>▶ 액터명 : 사용자</p> <p>▶ 유즈케이스 개요 및 설명 : 사용자는 노하우 글이나 커뮤니티 글에 댓글 작성을 할 수 있다.</p> <p>▶ 사전 조건 : 사용자는 로그인한 상태이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 노하우 글이나 커뮤니티 글을 조회한다.</p> <p>(2) 댓글 내용을 입력한다.</p> <p>(3) 등록 버튼을 누른다. (A1)</p> <p>- 대치흐름</p> <p>(A1) 댓글 내용을 입력하지 않은 경우. 댓글의 내용이 없음을 표시하고 내용 입력창으로 커서를 이동시킨다.</p>				

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김용훈	U-018
<p>▶ 유즈케이스명 : 공지사항 조회</p> <p>▶ 액터명 : 사용자</p> <p>▶ 유즈케이스 개요 및 설명 : 사용자는 공지사항을 조회할 수 있다.</p> <p>▶ 사전 조건 : 없음</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 사용자는 공지사항을 조회한다.</p>				

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김용훈	U-019

▶ 유즈케이스명 : FAQ 조회

▶ 액터명 : 사용자

▶ 유즈케이스 개요 및 설명 : 사용자는 FAQ를 조회할 수 있다.

▶ 사전 조건 : 없음

□ 이벤트 흐름

- 정상 흐름

(1) 사용자는 FAQ를 조회한다.

3) 유즈케이스 명세서 - 판매자

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김헌진	S-001

▶ 유즈케이스명 : 노하우 작성

▶ 액터명 : 판매자

▶ 유즈케이스 개요 및 설명 : 판매자가 노하우를 작성한다.

▶ 사전 조건 : 사용자는 판매자 등록이 된 상태이다.

□ 이벤트 흐름

- 정상 흐름

(1) 노하우 작성 버튼을 클릭한다.

(2) 제목, 소요비용, 유튜브 코드를 입력한다. (A1)

(3) 카테고리, 난이도, 소요시간, 소품 판매를 선택한다.

(4) 사진 추가버튼을 클릭하여 노하우 사진과 설명을 입력한다.

- 대치흐름

(A1) 유튜브 영상이 없는 경우 입력하지 않아도 된다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김헌진	S-002
<p>▶ 유즈케이스명 : 노하우 관리</p> <p>▶ 액터명 : 판매자</p> <p>▶ 유즈케이스 개요 및 설명 : 판매자가 작성한 노하우 글을 수정/삭제할 수 있다.</p> <p>▶ 사전 조건 : 사용자는 판매자 등록이 된 상태이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 판매자는 자신이 작성한 노하우 글을 조회한다.</p> <p>(2) 판매자는 노하우 글을 수정/삭제할 수 있다.</p>				

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김헌진	S-003
<p>▶ 유즈케이스명 : 판매한 내역 조회</p> <p>▶ 액터명 : 판매자</p> <p>▶ 유즈케이스 개요 및 설명 : 판매자가 판매한 내역을 조회한다.</p> <p>▶ 사전 조건 : 사용자는 판매자 등록이 된 상태이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 판매자는 자신이 판매한 인테리어 소품이나 재료의 내역을 조회한다.</p>				

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김헌진	S-004
<p>▶ 유즈케이스명 : 소품 발송</p> <p>▶ 액터명 : 판매자</p> <p>▶ 유즈케이스 개요 및 설명 : 판매자는 소품을 발송한다.</p> <p>▶ 사전 조건 : 판매자는 소품을 제작한 상태이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 판매자는 사용자에게 구매요청메시지를 받는다.</p> <p>(2) 판매자는 인테리어 소품을 제작한다.</p> <p>(3) 제작한 소품을 구매자에게 발송한다.</p> <p>(4) 발송 후 배송정보를 입력한다.</p>				

4) 유즈케이스 명세서 - 관리자

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김미선	M-001

▶ 유즈케이스명 : 판매자 등록관리

▶ 액터명 : 관리자

▶ 유즈케이스 개요 및 설명 : 관리자가 사용자의 계좌명의로 계좌번호 명의를 비교하여 같을 때 판매자로 등록한다.

▶ 사전 조건 : 사용자가 판매자 등록 신청을 한다.

□ 이벤트 흐름

- 정상 흐름
 - (1) 관리자는 사용자가 작성한 계좌명의로 계좌번호 명의를 확인한다.
 - (2) 계좌명의로 계좌번호 명의를 같은 경우 판매자로 등록을 한다. (A1)
- 대치흐름
 - (A1) 계좌명의로 계좌번호 명의를 같지 않은 경우 반려한다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김미선	M-002

▶ 유즈케이스명 : 공지사항 관리

▶ 액터명 : 관리자

▶ 유즈케이스 개요 및 설명 : 관리자는 사용자, 판매자, 재료 판매업체에게 공지할 사항을 작성한다.

▶ 사전 조건 : 없음

□ 이벤트 흐름

- 정상 흐름
 - (1) 관리자는 사용자, 판매자, 재료 판매업체에게 공지할 사항을 확인한다.
 - (2) 공지할 사항을 게시판에 작성한다.
 - (3) 작성한 글을 등록한다. (A1)
- 대치흐름
 - (A1) 관리자는 등록된 공지사항을 수정/삭제할 수 있다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김미선	M-003
<p>▶ 유즈케이스명 : FAQ 관리</p> <p>▶ 액터명 : 관리자</p> <p>▶ 유즈케이스 개요 및 설명 : 관리자는 FAQ를 작성한다.</p> <p>▶ 사전 조건 : 없음</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 관리자는 사용자, 판매자, 재료 판매업체가 자주 물어볼만한 질문들을 확인한다.</p> <p>(2) 질문과 답을 게시판에 작성한다.</p> <p>(3) 작성한 글을 등록한다. (A1)</p> <p>- 대치흐름</p> <p>(A1) 관리자는 등록된 FAQ를 수정/삭제할 수 있다.</p>				

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김미선	M-004
<p>▶ 유즈케이스명 : 거래 관리</p> <p>▶ 액터명 : 관리자</p> <p>▶ 유즈케이스 개요 및 설명 : 관리자는 구매자로부터 수령 확인 메시지를 확인한다.</p> <p>▶ 사전 조건 : 구매자는 판매자로부터 인테리어 소품을 수령한 상태이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 구매자는 인테리어 소품 수령 후 관리자에게 수령 확인 메시지를 전송한다.</p> <p>(2) 관리자는 사용자로부터 전송 받은 메시지를 확인한다. (A1)</p> <p>- 대치흐름</p> <p>(A1) 10일이 지나도 메시지가 오지 않을 경우 구매자에게 수령된 것으로 간주한다.</p>				

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김미선	M-005

▶ 유즈케이스명 : 푸쉬 알림

▶ 액터명 : 관리자

▶ 유즈케이스 개요 및 설명 : 관리자는 푸쉬 알림을 관리한다.

▶ 사전 조건 :

- (1) 구매자는 소품이나 재료를 구매하기 위해 결제를 한다. (결제 후 구매 요청 메시지가 자동으로 관리자에게 보내진다.)
- (2) 판매자, 재료판매업체는 배송정보를 입력한다.

□ 이벤트 흐름

- 정상 흐름
 - (1) 관리자는 구매자로부터 구매 요청 메시지를 확인한다. (A1),(A2)
 - (2) 관리자는 판매자나 재료 판매업체가 입력한 배송정보를 확인한다.
 - (3) 입력한 배송정보를 구매자에게 푸쉬 알림을 통해 알려준다.
- 대치흐름
 - (A1) 구매자가 소품을 구매하는 경우 판매자에게 푸쉬 알림을 통해 알려준다.
 - (A2) 구매자가 재료를 구매하는 경우 재료 판매업체에게 구매 요청 메시지가 보내진다.

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김미선	M-006

▶ 유즈케이스명 : 판매금액 송금

▶ 액터명 : 관리자

▶ 유즈케이스 개요 및 설명 : 관리자는 구매자로부터 수령 확인 메시지를 확인하여 판매자에게 수수료를 제외한 나머지 판매금액을 송금한다.

▶ 사전 조건 : 사용자는 판매자로부터 인테리어 소품을 수령한 상태이다.

□ 이벤트 흐름

- 정상 흐름
 - (1) 관리자는 구매자가 보낸 수령 확인 메시지를 확인한다. (A1)
 - (2) 메시지를 받은 경우 판매자에게 수수료를 제외한 나머지 판매금액을 송금한다.
- 대치흐름
 - (A1) 10일이 지나도 메시지가 오지 않을 경우 구매자에게 수령된 것으로 간주한다.

5) 유즈케이스 명세서 - 재료 판매업체

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김미선	IV-001
<p>▶ 유즈케이스명 : 재료 발송</p> <p>▶ 액터명 : 재료 판매업체</p> <p>▶ 유즈케이스 개요 및 설명 : 재료 판매업체는 재료를 발송한다.</p> <p>▶ 사전 조건 : 구매자가 재료 요청을 한 상태이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 재료 판매업체는 사용자에게 구매 요청 메시지를 받는다.</p> <p>(2) 재료를 구매자에게 발송한다.</p> <p>(3) 발송 후 배송정보를 입력한다.</p>				

6) 유즈케이스 명세서 - 배송업체

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김경희	DE-001
<p>▶ 유즈케이스명 : 위치정보 제공</p> <p>▶ 액터명 : 배송업체</p> <p>▶ 유즈케이스 개요 및 설명 : 주문한 제품에 대한 배송조회를 원하는 사용자에게 배송업체는 현재 택배의 위치 정보를 제공해준다.</p> <p>▶ 사전 조건 : 사용자가 제품을 주문한 상태이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 사용자가 배송위치 확인을 요청한다,</p> <p>(2) 배송업체가 사용자에게 위치정보를 제공한다.</p>				

7) 유즈케이스 명세서 - 결제 시스템

유즈케이스 명 세 서	프로젝트명	작성일	작성자	ID
	내가 꾸미는 우리 집 (내꾸우)	2016.04.24	김경희	PS-001
<p>▶ 유즈케이스명 : 결제</p> <p>▶ 액터명 : 결제시스템</p> <p>▶ 유즈케이스 개요 및 설명 : 사용자가 주문한 상품에 대해 결제한다.</p> <p>▶ 사전 조건 : 사용자가 주문할 상품을 결정한 상태이다.</p> <p>□ 이벤트 흐름</p> <p>- 정상 흐름</p> <p>(1) 사용자가 마음에 드는 상품을 선택한다.</p> <p>(2) 수량을 선택한다.</p> <p>(3) 결제 방식을 선택하고 요청한다.</p> <p>(4) 결제 시스템이 결제를 승인한다.</p>				

3. 시스템 설계서

가. 개요

1) 시스템의 목표

- 소품 판매자가 자신만의 인테리어 소품 판매 및 노하우를 공유할 수 있는 모바일 애플리케이션 제작
- 소품 구매자가 인테리어 소품의 재료 및 완제품을 구매할 수 있도록 결제시스템 구축

2) 시스템의 주요 기능

- 인테리어 소품과 재료의 판매 및 구매를 위한 결제 기능
- 사진과 동영상을 업로드 할 수 있는 노하우 작성 기능
- 거래 정보 관리, 배송 조회 및 부가 기능
- 댓글, 스크랩이 가능한 커뮤니티 기능

3) 설계상 제약 사항

- 결제시스템 연동이 불확실하다.
- 재료 판매업체와 협약을 맺는 것이 불확실하다.

나. 시스템 구조

1) 시스템 구조 개요

가) 클라이언트

클라이언트는 모바일 애플리케이션 내에서 노하우 조회, 재료 구매, 재료 조회, 완제품 구매, 검색, 커뮤니티, 글 등록, 소품판매 기능을 사용한다. 이러한 기능들을 서버에 요청하여 그 결과를 받는다.

나) 서버

클라이언트가 요청한 API를 통해 AWS S3와 DB에서 결과를 가져와 클라이언트에게 보낸다. API 종류로는 노하우API, 재료API, 회원API, 글API, 푸시API가 있다.

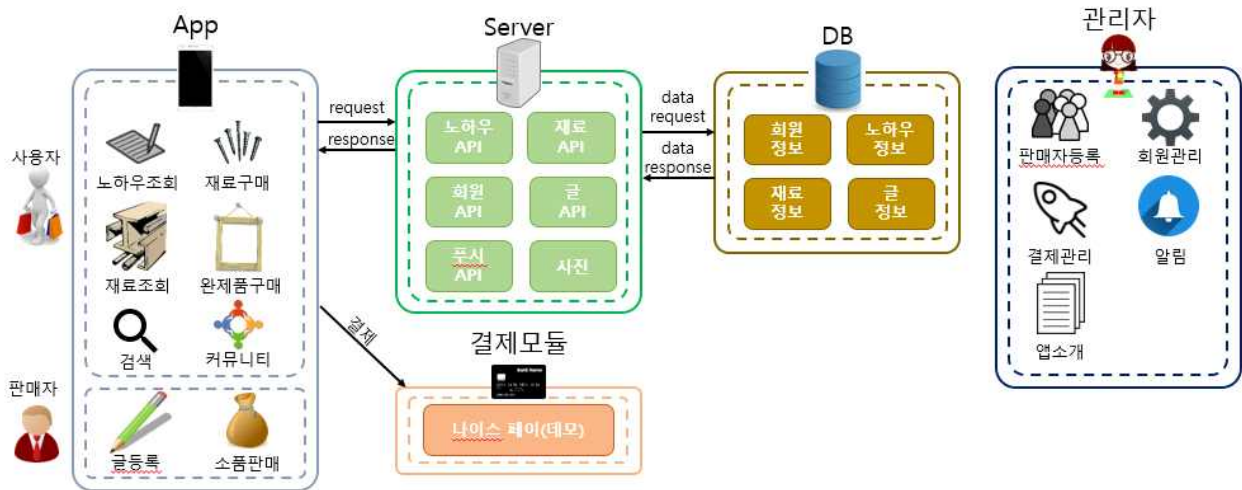
다) AWS S3 서버

AWS S3에는 사용자 정보와 사용자가 게시한 이미지와 동영상을 저장한다.

라) DB

DB에는 회원정보, 노하우 정보, 재료 정보, 글 정보 등을 저장한다.

2) 시스템 구조도



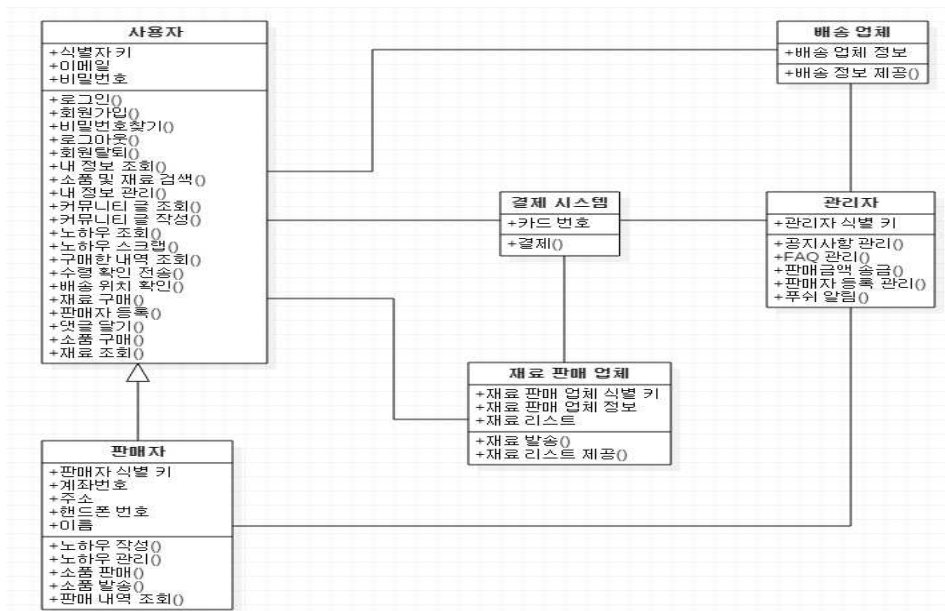
[그림 20] 시스템 구조도

다. 소프트웨어 설계

1) 클래스 다이어그램(class diagram)

클래스를 각 액터로 구별하여 필요 정보와 사용할 수 있는 기능들을 작성했다.

- 사용자는 애플리케이션을 사용하기 위한 기본 회원관련 기능 (로그인, 로그아웃, 내 정보 조회 등)과 애플리케이션에서 사용하는 기능 (노하우 조회, 재료 조회, 커뮤니티 글 조회 등)을 할 수 있다.
- 판매자는 사용자의 기능을 모두 포함하고 있으며, 노하우 작성과 관리가 가능하다.
- 결제시스템에는 카드 결제를 위한 결제 카드 번호가 변수로 들어간다.
- 관리자는 공지사항, FAQ, 판매금액 송금, 푸쉬 알림 등을 관리할 수 있다.
- 재료 판매 업체는 재료들의 리스트를 제공하고, 재료 구매 요청이 들어오면 재료를 배송한다.
- 배송업체는 사용자에게 배송 정보 조회를 해주기 위해 필요한 정보를 가지고 있다.

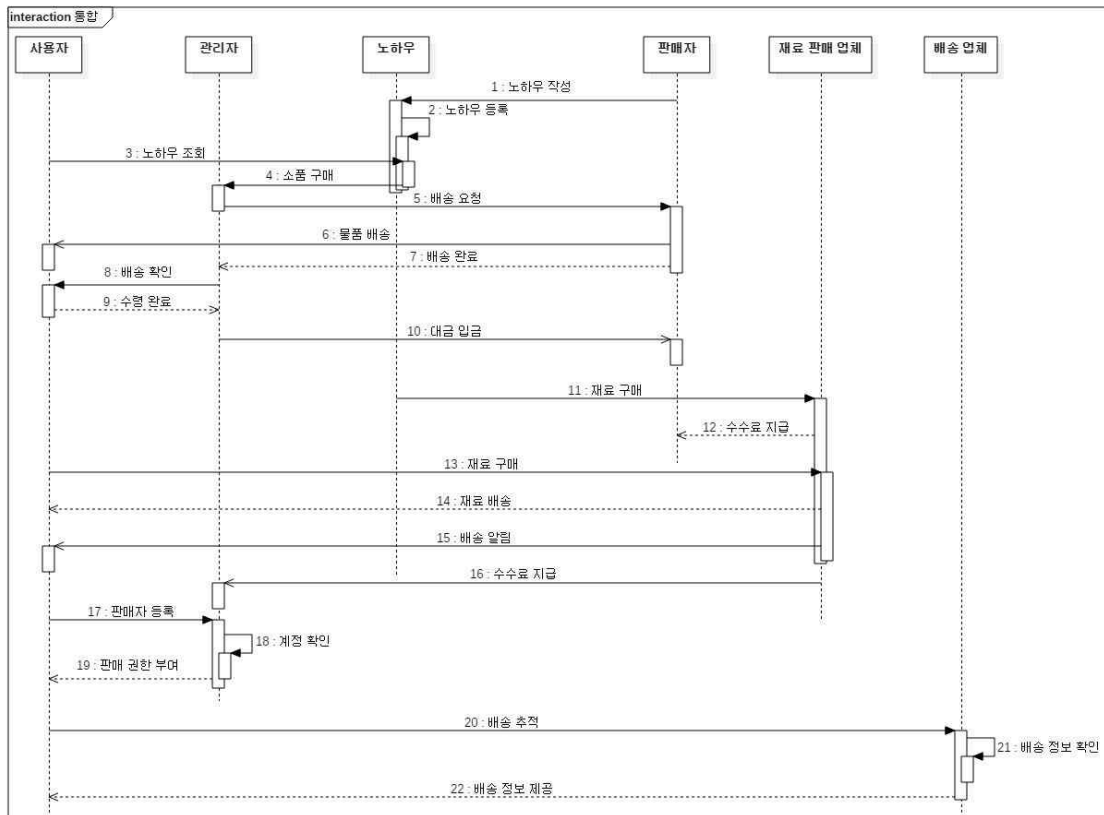


[그림 21] 클래스 다이어그램

2) 시퀀스 다이어그램(sequence diagram)

판매자가 노하우 글을 등록하면 사용자가 조회를 하고, 완제품이나 재료를 구매 후 배송조치를 하는 흐름을 나타낸다.

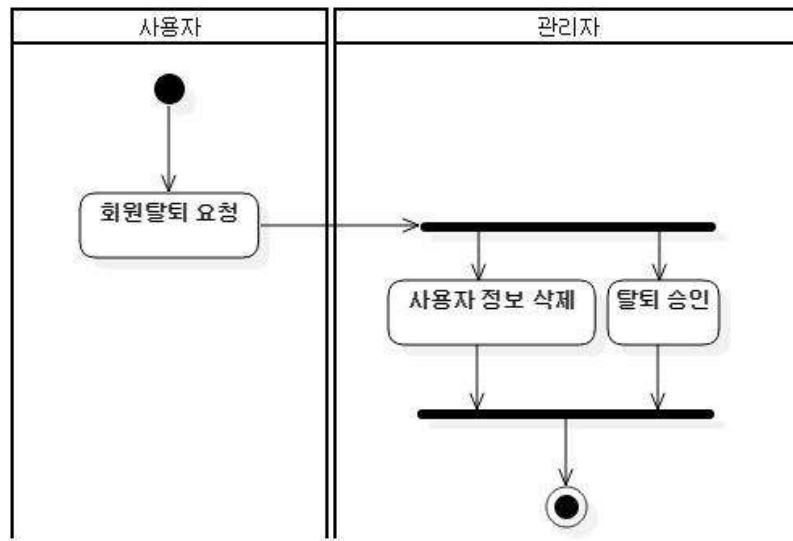
- 판매자는 인테리어 소품을 제작하는 노하우 글을 작성하면 노하우 글이 등록된다. (1)(2)
- 사용자는 노하우를 조회하고 완제품이나 재료를 구매할 수 있다. (3)(4)
- 사용자가 완제품 구매를 요청하면 관리자가 해당 판매자에게 배송을 요청한다. (5)
- 판매자는 사용자에게 물품을 배송하고 관리자에게 배송 완료를 알린다. (6)(7)
- 관리자는 사용자에게 배송 확인 메시지를 요청하면 사용자가 관리자에게 수령 완료 메시지를 보낸다. (8)(9)
- 관리자는 수령 완료 메시지를 확인하여 판매자에게 대금을 입금한다. (10)
- 사용자가 노하우 조회를 통해 노하우에 사용된 재료를 구매하는 경우 재료 판매 업체는 판매자에게 수수료를 지급한다. (11)(12)
- 사용자가 재료 구매를 요청하면 재료 판매 업체에서 재료를 배송하고 배송 알림 메시지를 보낸다. (13)(14)(15)
- 재료 판매 업체는 관리자에게 수수료를 지급한다. (16)
- 사용자가 인테리어 소품을 판매하기 위해 판매자 등록을 신청 하면 관리자가 계좌명의로 계좌번호 명의를 일치하는지 확인한 후 판매 권한을 부여한다. (17)(18)(19)
- 사용자는 완제품이나 재료를 구매하여 배송 업체에게 배송 추적을 요청하면 배송업체는 배송 정보를 확인하여 사용자에게 배송 정보를 제공한다. (20)(21)(22)



[그림 22] 시퀀스 다이어그램

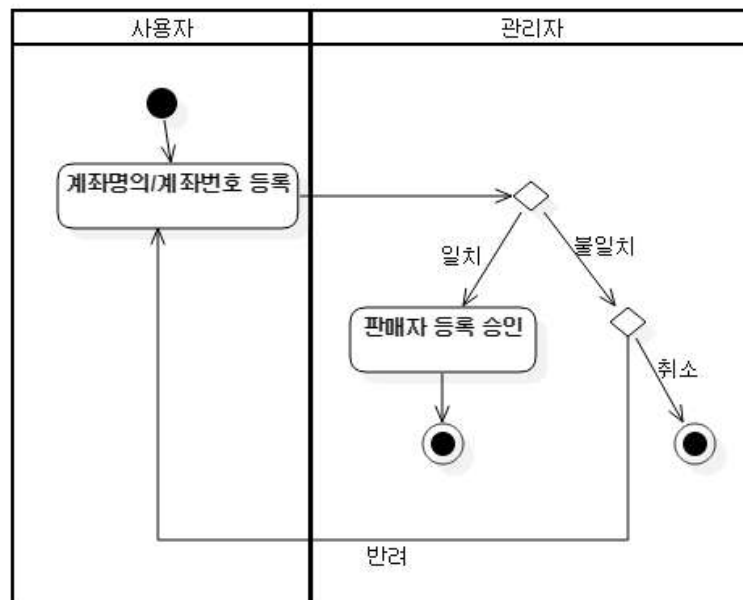
3) 활동 다이어그램(Activity Diagram)

가) 회원탈퇴



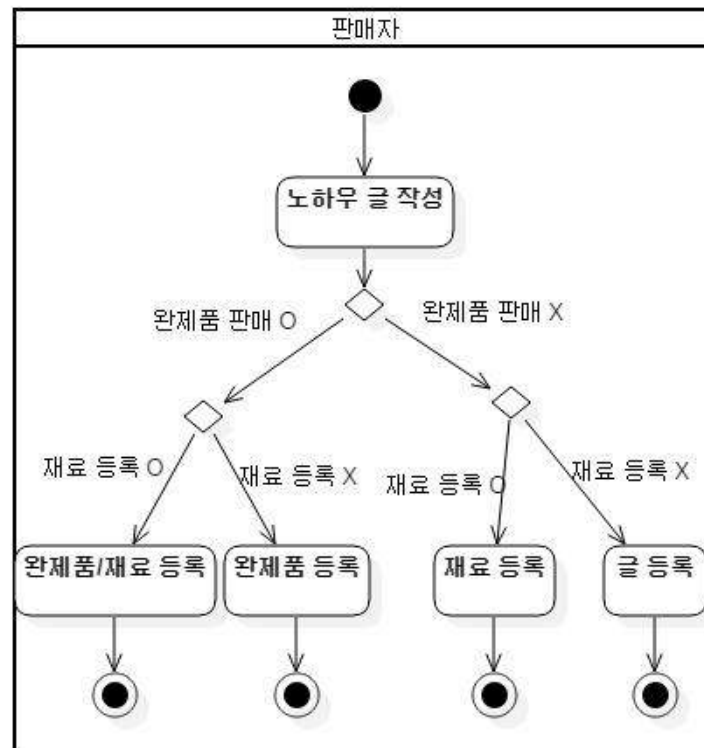
[그림 23] 회원탈퇴 활동 다이어그램

나) 판매자 등록



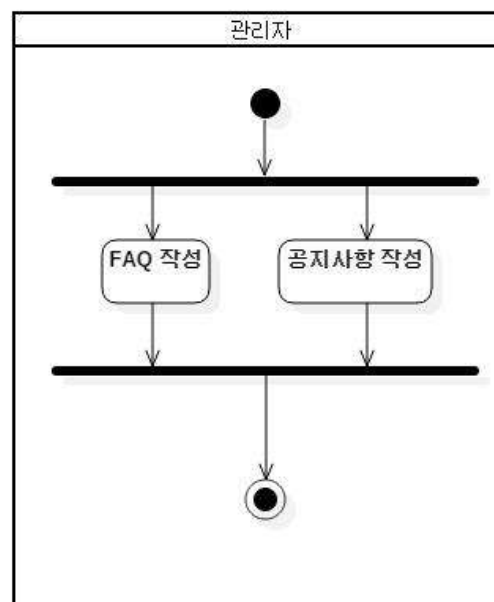
[그림 24] 판매자 등록 활동 다이어그램

다) 노하우 작성



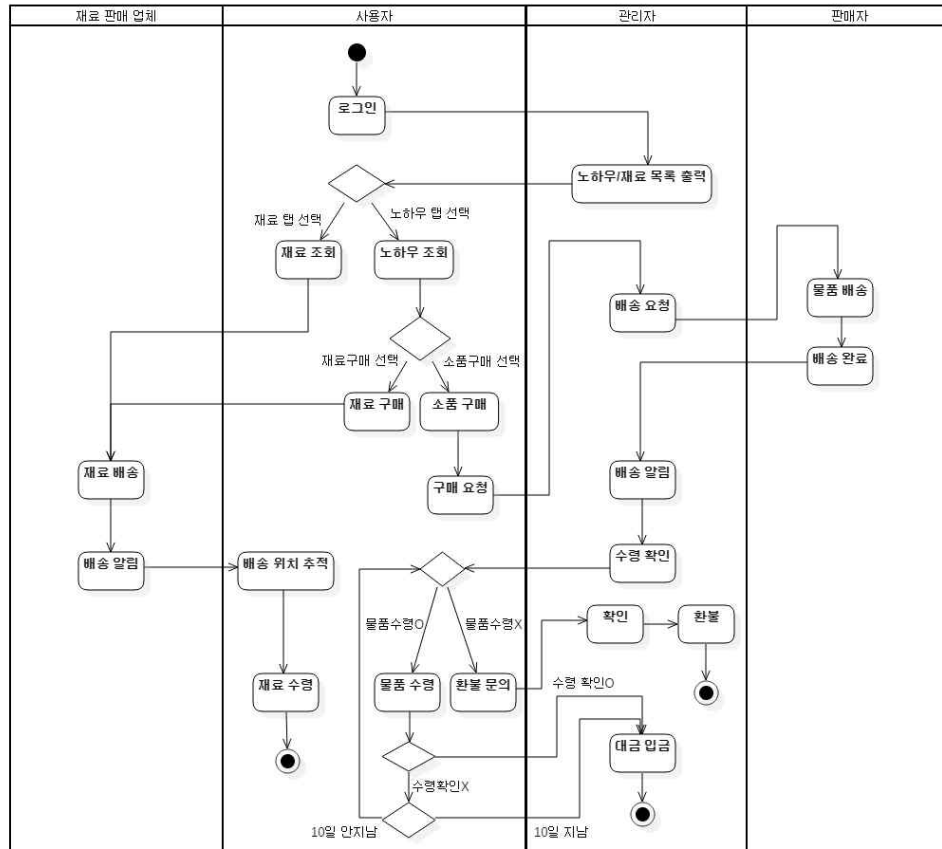
[그림 25] 노하우 작성 활동 다이어그램

라) 관리자



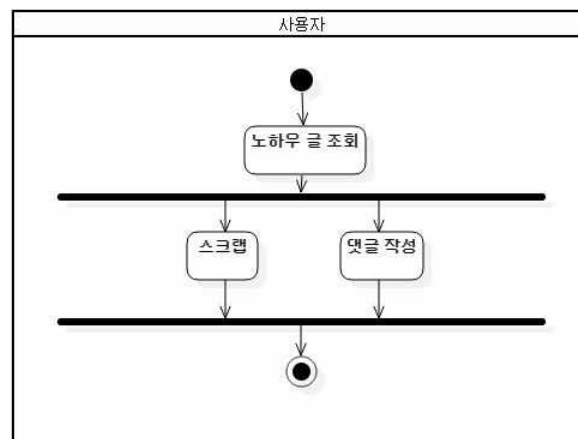
[그림 26] 관리자 활동 다이어그램

마) 소품, 재료 구매



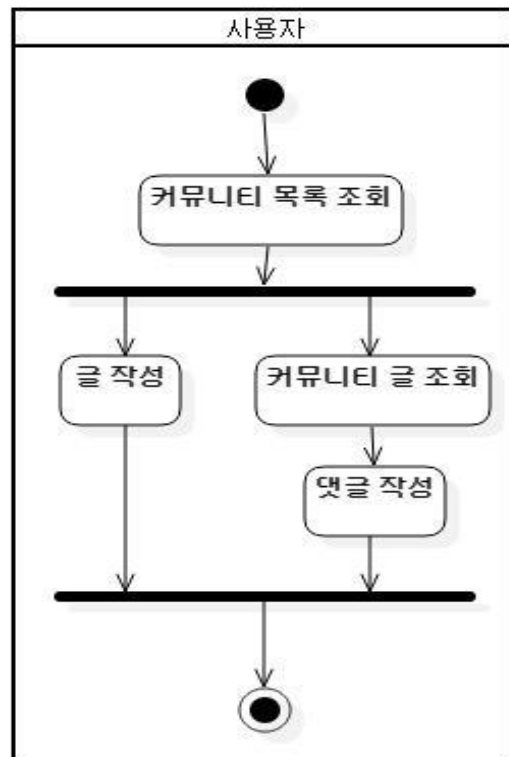
[그림 27] 소품, 재료 구매 활동 다이어그램

바) 스크랩



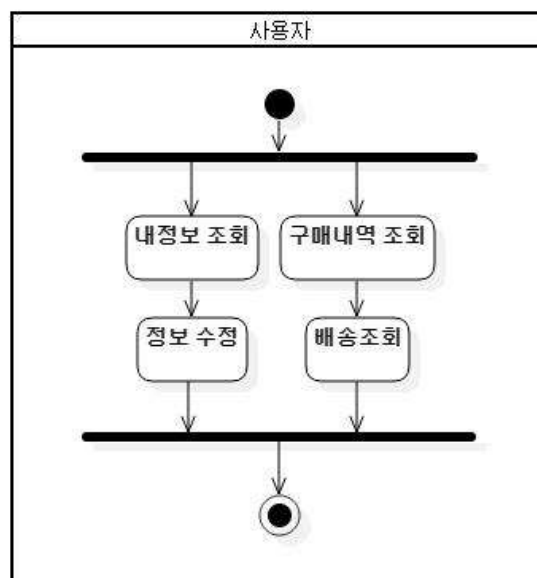
[그림 28] 스크랩 활동 다이어그램

사) 커뮤니티 글 작성



[그림 29] 커뮤니티 글 작성 활동 다이어그램

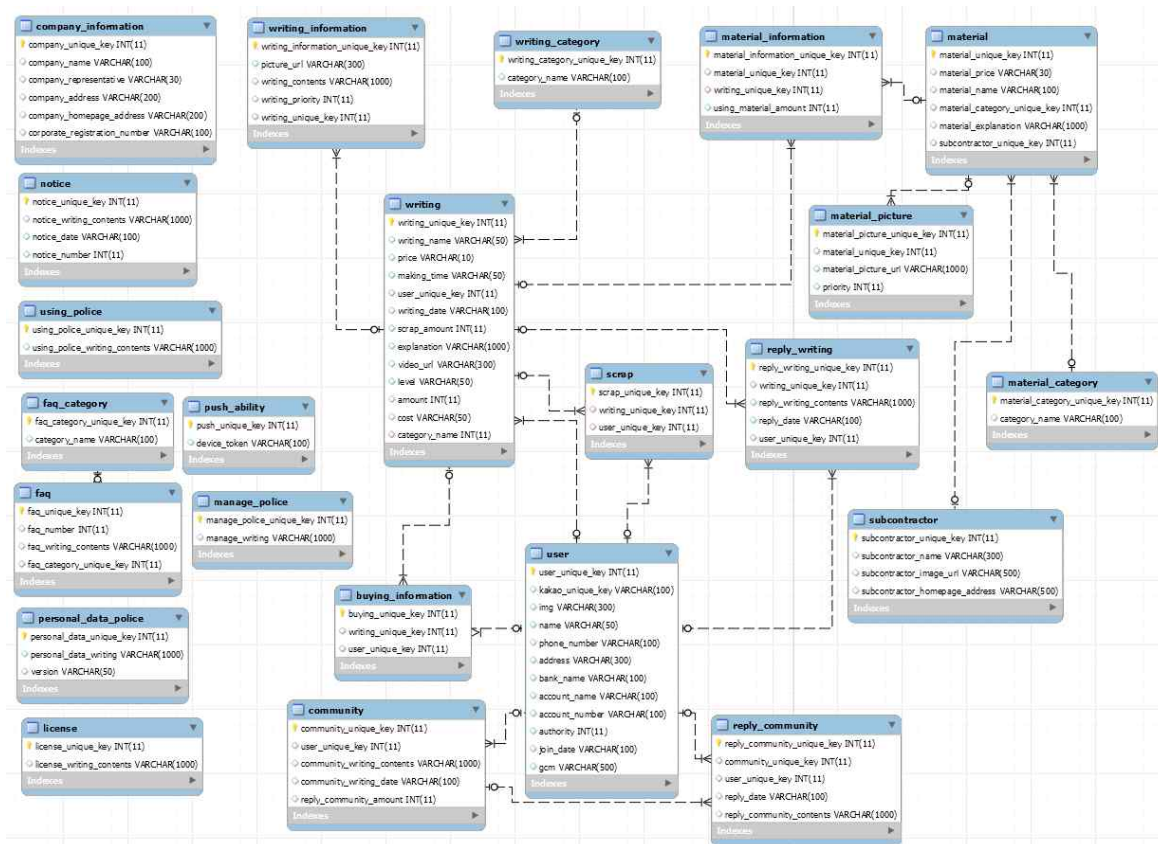
아) 내 정보 관리



[그림 30] 내 정보 관리 활동 다이어그램

라. 데이터베이스 설계

1) ER diagram



[그림 31] ER diagram

2) DB 테이블 기술서

Name		BUYING INFORMATION		Description		사용자의 구입 정보를 가지고 있다.
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN			
1	BUYING_UNIQUE_KEY	INTEGER	V	PK		구입 정보 고유번호
2	WRITING_UNIQUE_KEY	INTEGER		FK		노하우 글 고유번호
3	USER_UNIQUE_KEY	INTEGER		FK		사용자 고유번호
4	MATERIAL_UNIQUE_KEY	INTEGER		FK		재료 고유번호
비고						
WRITING_UNIQUE_KEY는 WRITING 테이블의 WRITING_UNIQUE_KEY를 참조하는 FK이다. USER_UNIQUE_KEY는 USER 테이블의 USER_UNIQUE_KEY를 참조하는 FK이다. MATERIAL_UNIQUE_KEY는 MATERIAL 테이블의 MATERIAL_UNIQUE_KEY를 참조하는 FK이다.						

Name		COMMUNITY	Description			커뮤니티에 대한 정보를 가지고 있다.	
System		내가 꾸미는 우리 집 (내꾸우)					
No	Column name	Data Type	NN	KY	Default		Description
1	COMMUNITY_UNIQUE_KEY	INTEGER	V	PK			커뮤니티 고유번호
2	USER_UNIQUE_KEY	INTEGER		FK			사용자 고유번호
3	COMMUNITY_WRITING_CONTENTS	VARCHAR(1000)					커뮤니티 글 내용
4	COMMUNITY_WRITING_DATE	VARCHAR(100)					커뮤니티 작성 날짜
5	REPLY_COMMUNITY_AMOUNT	INTEGER			0		커뮤니티 댓글 수
비고							
USER_UNIQUE_KEY는 USER 테이블의 USER_UNIQUE_KEY를 참조하는 FK이다.							

Name		COMPANY_INFORMATION		Description		내꾸우 애플리케이션을 제작한 회사에 대한 정보를 가지고 있다.
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN	KY	Default	Description
1	COMPANY_UNIQUE_KEY	INTEGER	V	PK		회사 고유번호
2	COMPANY_NAME	VARCHAR(100)				회사명
3	COMPANY_REPRESENTATIVE	VARCHAR(30)				회사 대표이름
4	COMPANY_ADDRESS	VARCHAR(200)				회사 주소
5	COMPANY_HOMEPAGE_ADDRESS	VARCHAR(200)				회사 홈페이지 주소
6	CORPORATE_REGISTRATION_NUMBER	VARCHAR(100)				사업자 등록번호
비고						

Name		FAQ	Description			FAQ에 대한 정보를 가지고 있다.
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type				
1	FAQ_UNIQUE_KEY	INTEGER	V	PK		FAQ 고유번호
2	FAQ_NUMBER	INTEGER				FAQ 글 번호
3	FAQ_WRITING_CONTENTS	VARCHAR(1000)				FAQ 글 내용
4	FAQ_CATEGORY_UNIQUE_KEY	INTEGER		FK		FAQ 카테고리 고유번호
비고						
FAQ_CATEGORY_UNIQUE_KEY는 FAQ_CATEGORY 테이블의 FAQ_CATEGORY_UNIQUE_KEY를 참조하는 FK이다.						

Name		FAQ_CATEGORY	Description			FAQ에 대한 카테고리 정보를 가지고 있다.	
System		내가 꾸미는 우리 집 (내꾸우)					
No	Column name	Data Type	NN	KY	Default	Description	
1	FAQ_CATEGORY_UNIQUE_KEY	INTEGER	V	PK		FAQ 카테고리 고유 번호	
2	CATEGORY_NAME	VARCHAR(100)				FAQ 카테고리 이름	
비고							

Name		LICENSE	Description			라이선스에 대한 정보를 가지고 있다.
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN	KY	Default	Description
1	LICENSE_UNIQUE_KEY	INTEGER	V	PK		라이선스 고유번호
2	LICENSE_WRITING_CONTENTS	VARCHAR(1000)				라이선스 글 내용
비고						

Name		MANAGE_POLICE	Description			운영정책에 대한 정보를 가지고 있다.	
System		내가 꾸미는 우리 집 (내꾸우)					
No	Column name	Data Type					
1	MANAGE_POLIC E_UNIQUE_KE Y	INTEGER	V	PK		운영정책 고유번호	
2	MANAGE_WRITI NG	VARCHAR(1 000)				운영정책 내용	
비고							

Name		MATERIAL	Description			재료에 대한 정보를 가지고 있다.
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN	KY	Default	Description
1	MATERIAL_UNIQUE_KEY	INTEGER	V	PK		재료 고유번호
2	MATERIAL_PRICE	VARCHAR(30)				재료 가격
3	MATERIAL_NAME	VARCHAR(100)				재료 이름
4	MATERIAL_CATEGORY_UNIQUE_KEY	INTEGER		FK		재료 카테고리
5	MATERIAL_EXPLANATION	VARCHAR(1000)				재료 설명
6	SUBCONTRACTOR_UNIQUE_KEY	INTEGER		FK		협약업체 고유번호
비고						
MATERIAL_CATEGORY_UNIQUE_KEY는 MATERIAL_CATEGORY 테이블의 MATERIAL_CATEGORY_UNIQUE_KEY를 참조하는 FK이다. SUBCONTRACTOR_UNIQUE_KEY는 SUBCONTRACTOR 테이블의 SUBCONTRACTOR_UNIQUE_KEY를 참조하는 FK이다.						

Name		MATERIAL_CATEGORY	Description			재료의 카테고리 정보를 가지고 있다.
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN	KY	Default	Description
1	MATERIAL_CATEGORY_UNIQUE_KEY	INTEGER	V	PK		재료 카테고리 고유 번호
2	CATEGORY_NAME	VARCHAR(100)				재료 카테고리 이름
비고						

Name		MATERIAL_INFOR MATION	Description			노하우에 사용된 재료의 정보를 가지고 있다.
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type				
1	MATERIAL_INFO RMATION_UN IQUE_KEY	INTEGER	V	PK		재료 정보 고유번호
2	MATERIAL_UNIQ UE_KEY	INTEGER		FK		재료 고유번호
3	WRITING_UNIQ UE_KEY	INTEGER		FK		노하우 글 고유번호
4	USING_MATERIA L_AMOUNT	INTEGER				사용된 재료 개수
비고						
MATERIAL_UNIQUE_KEY는 MATERIAL 테이블의 MATERIAL_UNIQUE_KEY를 참조한 FK이다. WRITING_UNIQUE_KEY는 WRITING 테이블의 WRITING_UNIQUE_KEY를 참조한 FK이다. SUBCONTRACTOR_UNIQUE_KEY는 SUBCONTRACTOR 테이블의 SUBCONTRACTOR_UNIQUE_KEY를 참조한 FK이다.						

Name		MATERIAL_PICTURE	Description			재료 사진에 대한 정보를 가지고 있다.
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type				
1	MATERIAL_PICTURE_UNIQUE_KEY	INTEGER	V	PK		재료 사진 고유번호
2	MATERIAL_UNIQUE_KEY	INTEGER		FK		재료 고유번호
3	MATERIAL_PICTURE_URL	VARCHAR(1000)				재료 사진 URL
4	PRIORITY	INTEGER				우선순위
비고						
MATERIAL_UNIQUE_KEY는 MATERIAL 테이블의 MATERIAL_UNIQUE_KEY를 참조한 FK이다.						

Name		NOTICE	Description			관리자가 작성한 공지사항에 대한 정보를 가지고 있다.
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN	KY	Default	Description
1	NOTICE_UNIQUE_KEY	INTEGER	V	PK		공지사항 고유번호
2	NOTICE_WRITING_CONTENTS	VARCHAR(1000)				공지사항 글 내용
3	NOTICE_DATE	VARCHAR(100)				공지사항 작성 날짜
4	NOTICE_NUMBER	INTEGER				공지사항 글 번호
비고						

Name		PERSONAL_DATA_POLICE		Description		개인정보 취급방침에 대한 정보를 가지고 있다.	
System		내가 꾸미는 우리 집 (내꾸우)					
No	Column name	Data Type	NN	KY	Default	Description	
1	PERSONAL_DATA_UNIQUE_KEY	INTEGER	V	PK		개인정보 취급방침 고유번호	
2	PERSONAL_DATA_WRITING	VARCHAR(1000)				개인정보 취급방침 내용	
3	VERSION	VARCHAR(50)				버전	
비고							

Name		PUSH_ABILITY		Description		푸쉬 알림을 하기 위한 장치 정보를 가지고 있다.	
System		내가 꾸미는 우리 집 (내꾸우)					
No	Column name	Data Type	NN				
1	PUSH_UNIQUE_KEY	INTEGER	V	PK		푸쉬 알림 고유번호	
2	DEVICE_TOKEN	VARCHAR(100)				토큰값	
비고							

Name		REPLY_COMMUNITY		Description		커뮤니티 댓글에 대한 정보를 가지고 있다.
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN	KY	Default	Description
1	REPLY_COMMUNITY_UNIQUE_KEY	INTEGER	V	PK		커뮤니티 댓글 고유번호
2	COMMUNITY_UNIQUE_KEY	INTEGER		FK		커뮤니티 고유번호
3	USER_UNIQUE_KEY	INTEGER		FK		댓글 작성한 사용자 고유번호
4	REPLY_DATE	VARCHAR(100)				댓글 작성 날짜
5	REPLY_COMMUNITY_CONTENTS	VARCHAR(1000)				커뮤니티 댓글 내용
비고						
COMMUNITY_UNIQUE_KEY는 COMMUNITY 테이블의 COMMUNITY_UNIQUE_KEY를 참조하는 FK이다.						
USER_UNIQUE_KEY는 USER 테이블의 USER_UNIQUE_KEY를 참조하는 FK이다.						

Name		REPLY_WRITING	Description			노하우 댓글 작성에 대한 정보를 가지고 있다.
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN	KY	Default	Description
1	REPLY_WRITING_UNIQUE_KEY	INTEGER	V	PK		댓글 작성 고유 번호
2	WRITING_UNIQUE_KEY	INTEGER		FK		노하우 글 고유번호
3	REPLY_WRITING_CONTENTS	VARCHAR(1000)				노하우 댓글 내용
4	REPLY_DATE	VARCHAR(100)				댓글 작성 날짜
5	USER_UNIQUE_KEY	INTEGER		FK		사용자 고유번호
비고						
WRITING_UNIQUE_KEY는 WRITING테이블의 WRITING_UNIQUE_KEY를 참조하는 FK이다. USER UNIQUE KEY는 USER테이블의 USER UNIQUE KEY를 참조하는 FK이다.						

Name		SCRAP		Description		사용자가 스크랩한 노하우 글에 대한 정보를 가지고 있다.	
System		내가 꾸미는 우리 집 (내꾸우)					
No	Column name	Data Type	NN	KY	Default		Description
1	SCRAP_UNIQUE_KEY	INTEGER	V	PK			스크랩 고유번호
2	WRITING_UNIQUE_KEY	INTEGER		FK			노하우 글 고유번호
3	USER_UNIQUE_KEY	INTEGER		FK			사용자 고유번호
비고							
WRITING_UNIQUE_KEY는 WRITING테이블의 WRITING_UNIQUE_KEY를 참조하는 FK이다. USER_UNIQUE_KEY는 USER테이블의 USER_UNIQUE_KEY를 참조하는 FK이다.							

Name		SUBCONTRACTOR	Description			협약업체에 대한 정보를 가지고 있다.
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN	KY	Default	Description
1	SUBCONTRACTOR_UNIQUE_KEY	INTEGER	V	PK		협약업체 고유번호
2	SUBCONTRACTOR_NAME	VARCHAR(300)				협약업체 회사명
3	SUBCONTRACTOR_IMAGE_URL	VARCHAR(500)				협약업체 이미지 URL
4	SUBCONTRACTOR_HOMEPAGE_ADDRESS	VARCHAR(500)				협약업체 홈페이지 주소
비고						

Name		USER	Description			사용자에 대한 정보를 가지고 있다.
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN	KY	Default	Description
1	USER_UNIQUE_KEY	INTEGER	V	PK		사용자 고유번호
2	KAKAO_UNIQUE_KEY	VARCHAR(100)				카카오 고유번호
3	IMG	VARCHAR(300)				사용자 이미지
4	NAME	VARCHAR(50)				이름
5	PHONE_NUMBER	VARCHAR(100)				전화번호
6	ADDRESS	VARCHAR(300)				주소
7	BANK_NAME	VARCHAR(100)				은행명
8	ACCOUNT_NAME	VARCHAR(100)				계좌명의
9	ACCOUNT_NUMBER	VARCHAR(100)				계좌번호
10	AUTHORITY	INTEGER			0	판매자 권한
11	JOIN_DATE	VARCHAR(100)				가입 날짜
12	GCM	VARCHAR(500)			1	푸쉬 알림
비고						

Name		USING_POLICE	Description		이용약관에 대한 정보를 가지고 있다.	
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN	KY	Default	Description
1	USING_POLICE_UNIQUE_KEY	INTEGER	V	PK		이용약관 고유번호
2	USING_POLICE_WRITING_CONTENTS	VARCHAR(1000)				이용약관 내용
비고						

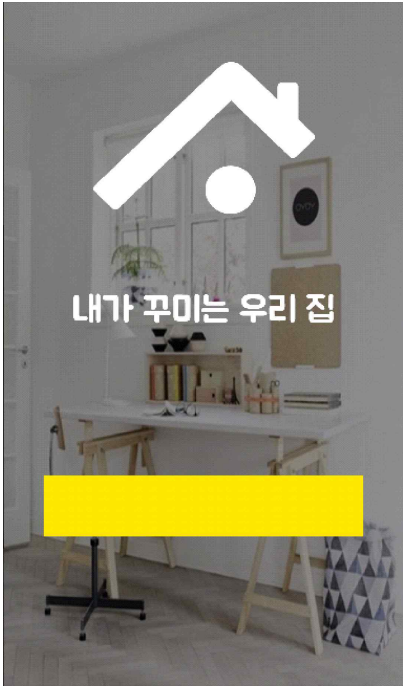
Name		WRITING	Description		노하우에 대한 정보를 가지고 있다.	
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN	KY	Default	Description
1	WRITING_UNIQUE_KEY	INTEGER	V	PK		노하우 글 고유번호
2	WRITING_NAME	VARCHAR(50)				노하우 글 제목
3	PRICE	VARCHAR(10)				판매비용
4	MAKING_TIME	VARCHAR(50)				제작 시간
5	USER_UNIQUE_KEY	INTEGER		FK		사용자 고유번호
6	WRITING_DATE	VARCHAR(100)				작성 날짜
7	SCRAP_AMOUNT	INTEGER			0	스크랩 수
8	CATEGORY_NAME	VARCHAR(100)		FK		카테고리 이름
9	EXPLANATION	VARCHAR(1000)				노하우 설명
10	VIDEO_URL	VARCHAR(300)				동영상 URL
11	LEVEL	VARCHAR(50)				난이도
12	AMOUNT	INTEGER				판매할 개수
13	COST	VARCHAR(50)				소요 비용
비고						
USER_UNIQUE_KEY는 USER테이블의 USER_UNIQUE_KEY를 참조하는 FK이다. CATEGORY_NAME는 CATEGORY 테이블의 CATEGORY_NAME을 참조하는 FK이다.						

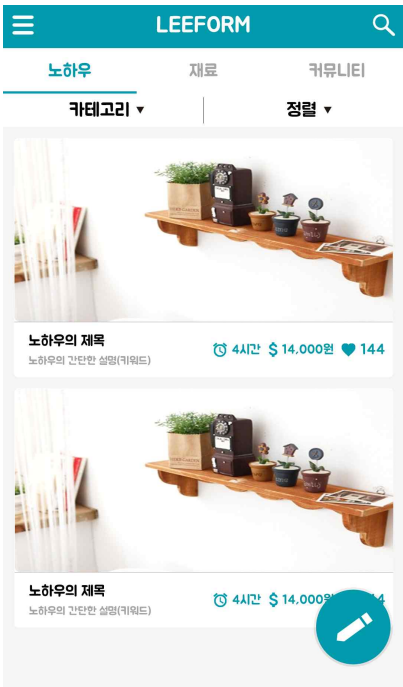
Name		WRITING_CATEGOR	Description		노하우에 대한 카테고리 정보를 가지고 있다.	
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN	KY	Default	Description
1	WRITING_CATEGORY_UNIQUE_KEY	INTEGER	V	PK		노하우 카테고리 고유번호
2	CATEGORY_NAME	VARCHAR(100)				카테고리 이름
비고						

Name		WRITING_INFORMATION	Description		노하우의 제작과정(사진과 설명)에 대한 정보를 가지고 있다.	
System		내가 꾸미는 우리 집 (내꾸우)				
No	Column name	Data Type	NN	KY	Default	Description
1	WRITING_INFORMATION_UNIQUE_KEY	INTEGER	V	PK		노하우 정보 고유번호
2	PICTURE_URL	VARCHAR(300)				사진 URL
3	WRITING_CONTENTS	VARCHAR(1000)				글 내용(노하우 설명)
4	WRITING_PRIORITY	INTEGER				작성 글 우선순위
5	WRITING_UNIQUE_KEY	INTEGER		FK		작성 글 고유번호
비고						
WRITING_UNIQUE_KEY는 WRITING테이블의 WRITING_UNIQUE_KEY를 참조하는 FK이다.						

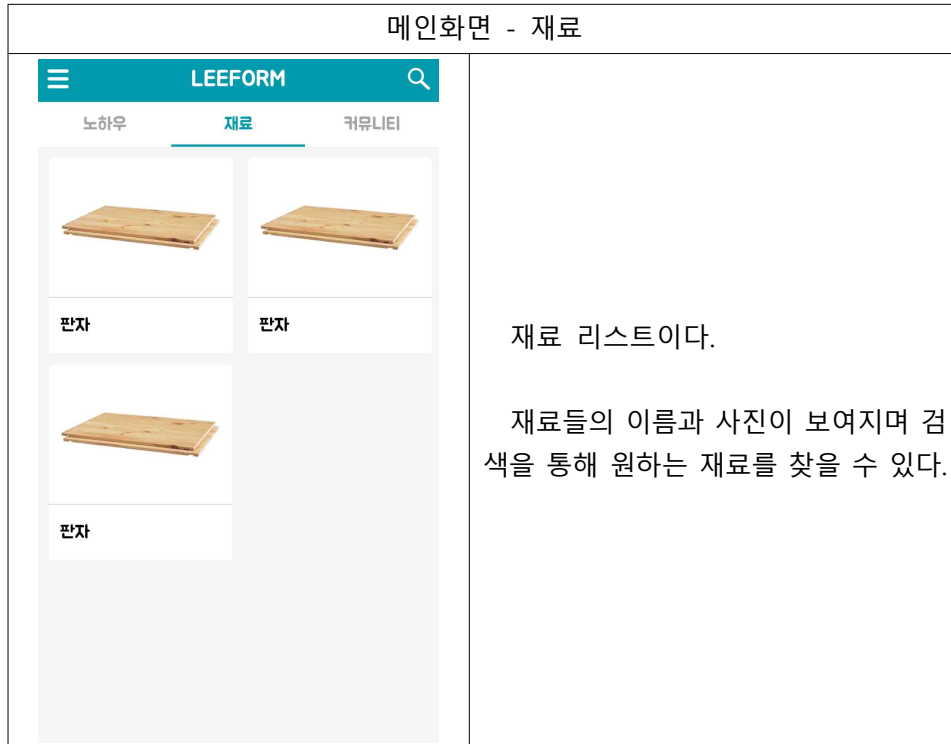
마. 사용자 인터페이스

1) UI 설계서

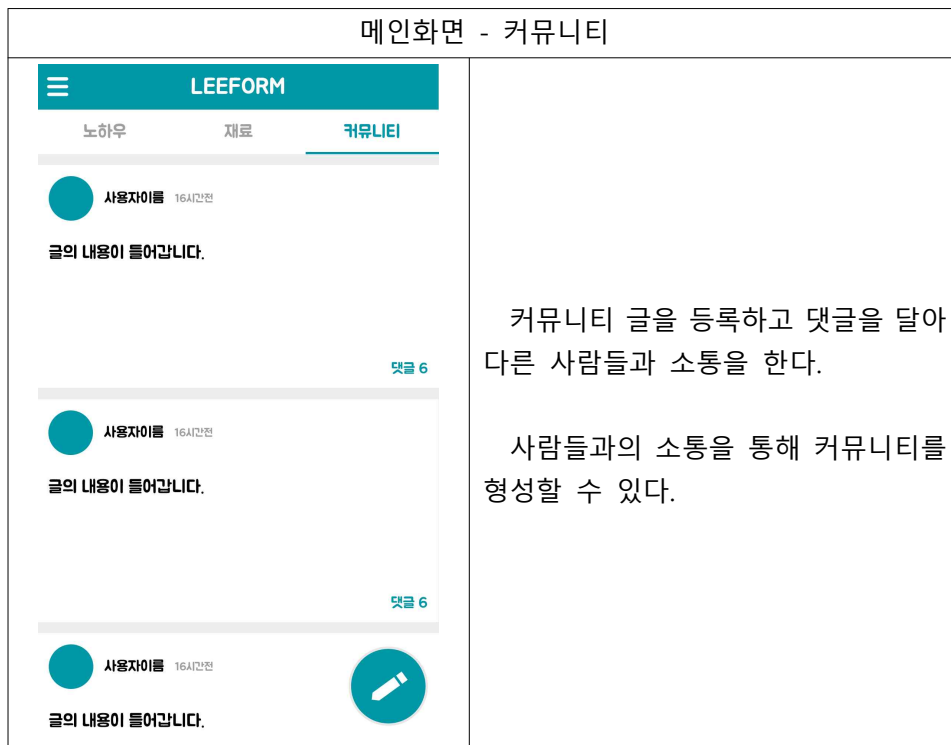
로그인	
	<p>카카오톡 로그인과 연동하여 로그인 한다.</p> <p>한 번 로그인을 하게 되면 다음에 애플리케이션을 실행할 때 로그인 화면이 아닌 메인 페이지로 넘어가게 된다.</p>

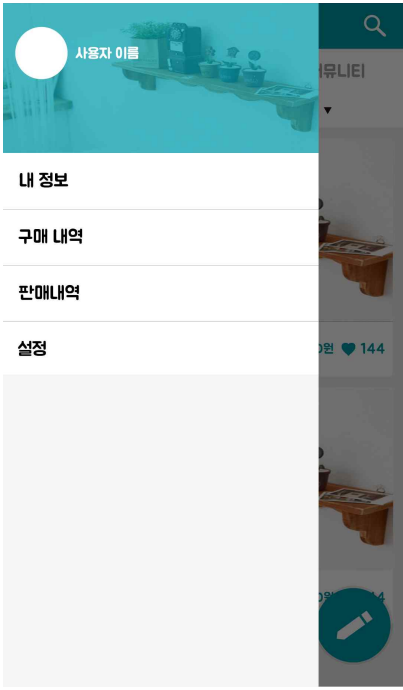
메인화면 - 노하우	
	<p>로그인을 성공했을 때 보이는 메인 화면이다. 판매자들이 작성한 노하우들의 리스트를 보여주며 카테고리화 최신순 등을 이용하여 정렬이 가능하다.</p> <p>검색을 통해 원하는 노하우를 찾을 수 있다.</p> <p>슬라이드를 통해 재료, 커뮤니티 탭으로 이동을 할 수 있다.</p>

메인화면 - 재료



메인화면 - 커뮤니티



네비게이션	
	<p>네비게이션에 들어가는 내용들로 내 정보, 구매 내역, 판매내역, 설정 등을 확인할 수 있다.</p> <p>사용자의 권한이 판매자인 경우 판매 내역 항목으로 표시되고 판매자가 아닌 경우에는 판매자등록 항목으로 표시된다.</p>

노하우 작성	
	<p>노하우 작성 페이지이다.</p> <p>노하우 작성을 위해 제목, 소모비용을 입력하며, 카테고리, 소요시간, 난이도를 체크한다.</p> <p>유튜브 영상이 있을 경우 코드를 입력한다.</p> <p>갤러리에 있는 사진을 찾아 사진과 사진에 대한 설명글을 등록한다. '+' 버튼을 눌러 여러 개의 사진과 글을 등록할 수 있다.</p> <p>소품판매를 원하는 경우 체크하여 판매수량과 판매가격을 입력한다.</p>

노하우 조회



노하우 탭의 리스트 중 하나를 클릭하면 나오는 화면이다.

유튜브 동영상이 있을 경우 동영상이 보여진다.

노하우 제목, 설명, 작성자, 제작정보, 제작순서 등이 보인다.

작성자가 완제품을 판매하는 경우 완제품을 구매할 수 있다.

사용자가 마음에 드는 노하우를 스크랩할 수 있다.

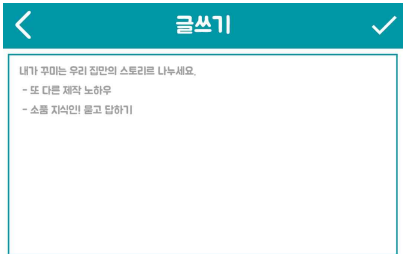
재료 조회




재료 탭의 리스트 중 하나를 클릭하면 나오는 화면이다.

재료의 사진들과 재료 이름과 가격이 보인다.

판매처 정보를 알 수 있으며 재료를 구매할 수 있다.

커뮤니티 작성	
 <p>Community Write Screen Mockup: A teal header bar with a back arrow, '글쓰기' (Write), and a checkmark. Below is a text input area with placeholder text: '내가 꾸미는 우리 집안의 스토리를 나누세요. ~ 또 다른 재작 노하우 ~ 소름 지식만 들고 답하기'.</p>	<p>메인 페이지의 커뮤니티에서 커뮤니티 글을 작성하는 페이지다.</p>

커뮤니티 조회	
 <p>Community List Screen Mockup: A teal header bar with a back arrow. Below is a user profile section with a teal circle, '사용자이름', and '16시간 전'. The main content area shows a list of community posts, each with a teal circle, '사용자이름', '댓글내용', and '11시간 전'. At the bottom is a text input field with '댓글로 표현하세요!' and a '등록' (Post) button.</p>	<p>커뮤니티 탭의 리스트 중 하나를 클릭하면 나오는 화면이다.</p> <p>커뮤니티 작성자 이름과 글의 내용이 있으며 글에 대한 댓글이 보인다.</p> <p>사용자는 커뮤니티 글에 댓글을 달 수 있다.</p>

결제화면 - 소품	
<div> <div> <div>←</div> <div>LEEFORM</div> </div> <div> <div> <div>사진</div> <div>소품 이름</div> <div>가격 <input type="text"/></div> <div>수량 <input type="text"/></div> </div> <div> <div>총 가격</div> <div>원</div> </div> </div> <div> <div> <div>판매자 정보</div> <div>이름</div> <div>판매자 주소</div> </div> <div> <div>구매자 정보</div> <div>이름</div> <div>구매자 주소</div> </div> <div>결제 완료</div> </div> </div>	<p>소품을 결제하는 화면으로 수량을 입력할 수 있다.</p> <p>판매자와 구매자에 대한 간략한 정보가 나오며, 구매자는 구매하려고 하는 사용자의 정보이다.</p>

결제화면 - 재료	
<div> <div> <div>←</div> <div>LEEFORM</div> </div> <div> <div> <div>사진</div> <div>재료 이름</div> <div>가격 <input type="text"/></div> <div>수량 <input type="text"/></div> </div> <div> <div>+</div> </div> </div> <div> <div>총 가격:</div> <div>원</div> <div>결제 완료</div> </div> </div>	<p>재료를 구매하는 페이지로 메인 리스트에서 재료를 구매하는 것과 노하우 내에서 재료 구매하는 방법 2가지가 있다.</p> <p>메인 리스트에서 재료를 구매할 경우 추가를 통해 원하는 재료들을 한 번에 결제할 수 있으며, 노하우 리스트에서 재료를 구매할 경우 사용된 재료들이 미리 추가가 된 상태로 나온다.</p> <p>필요 시 '+' 버튼을 통해 원하는 재료를 추가할 수 있다.</p>

4. 코딩보고서

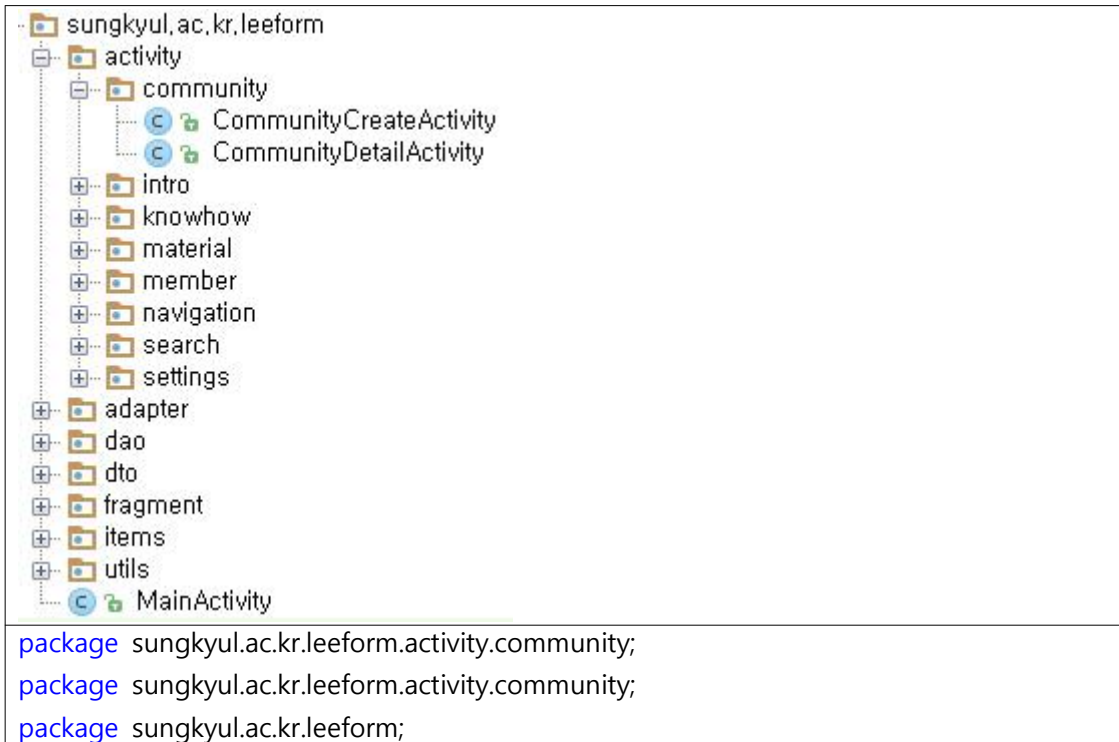
가. 프로그램 명명 규칙

- 주로 소문자 위주로 사용하며, 복합적인 단어의 경우 두 번째 단어부터 분리의 표시로 대문자를 사용하여 혼돈을 줄인다.
- 메소드나 변수, 또는 파라미터 이름의 맨 앞줄에 나오는 약자는 특수한 경우를 제외하고 소문자로 한다.

```
private int mNumber;  
private String mCost;  
private String mTime;  
private String mLike;  
private int mImg;  
private String mUrl;
```

1) 패키지 이름

- 기본 패키지 이름은 sungkyul.ac.kr.leeform이다.
- 패키지는 각 클래스 혹은 하위 패키지들의 집합이다.
- 유사한 기능의 클래스를 하나로 묶을 수 있게 이름을 정한다.
- 패키지는 소문자로 시작한다. (대문자로 할 시 프로젝트의 버그가 생길 수 있다.)

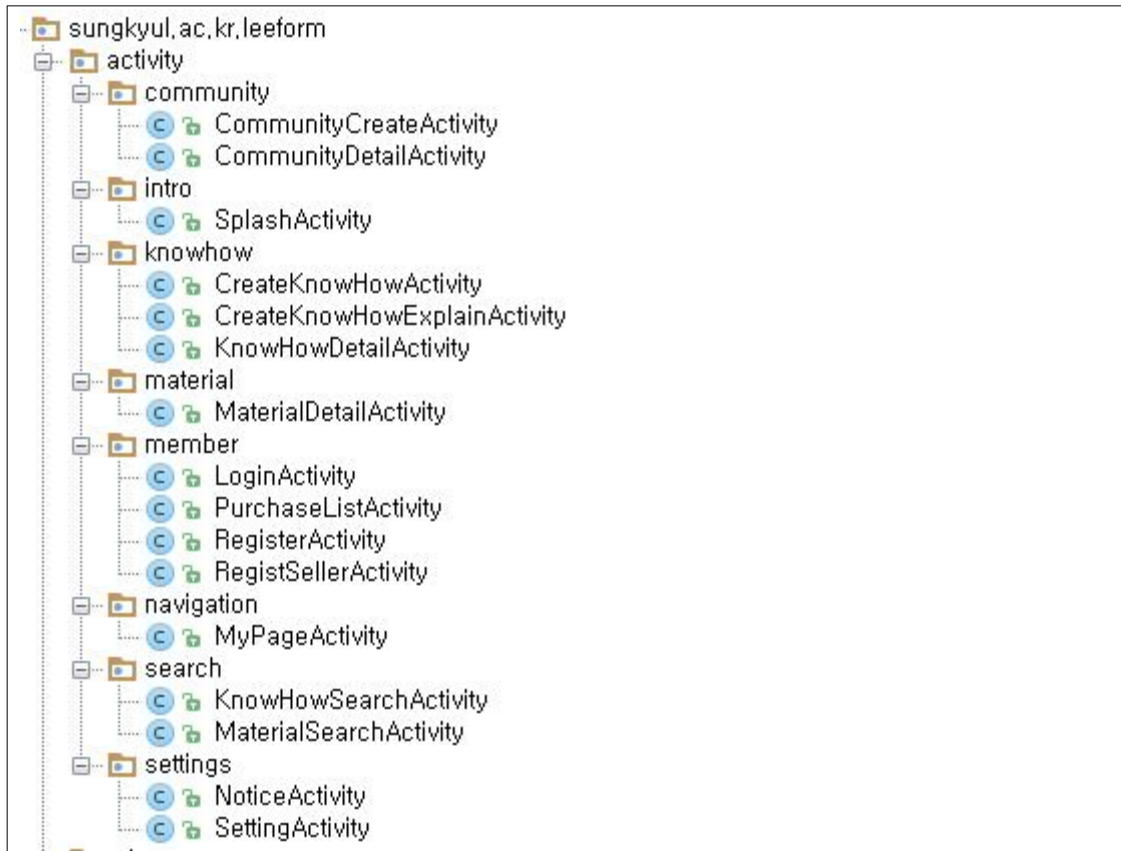


```
package sungkyul.ac.kr.leeform.activity.community;  
package sungkyul.ac.kr.leeform.activity.community;  
package sungkyul.ac.kr.leeform;
```

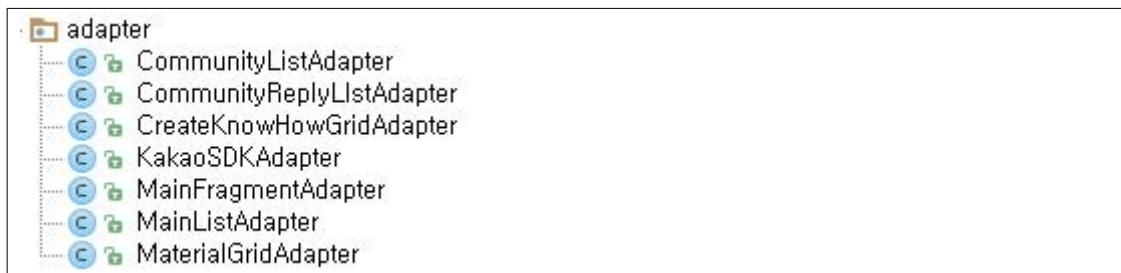
2) 타입 이름

- 클래스는 Pascal Casing 표기법을 따른다. (대문자 시작, 구분되는 단어 대문자)
- 클래스의 이름을 정의 할 때는 명사를 사용한다.
- 클래스의 이름을 보고 어떠한 기능을 하는지 알 수 있게 한다.

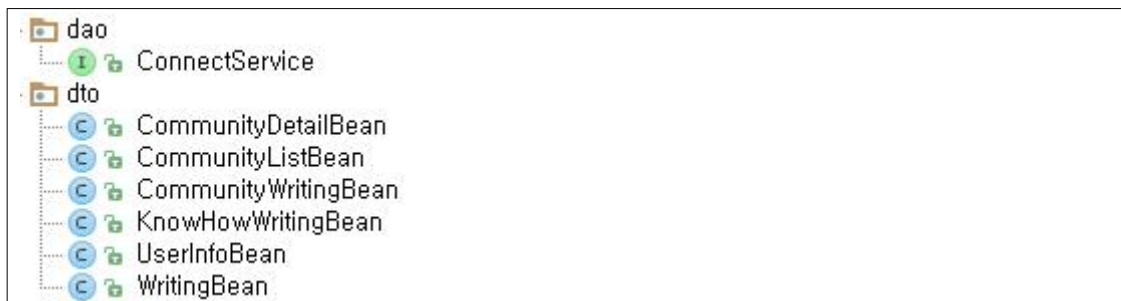
- 클래스가 Activity의 기능을 수행하면 마지막에 Activity를 붙인다.



- 클래스가 Adapter의 기능을 수행하면 마지막에 Adapter를 붙인다.



- 클래스가 DB의 기능을 수행하면 마지막에 Service, Bean을 붙인다.



- 클래스가 Fragment의 기능을 수행하면 마지막에 Fragment를 붙인다.



- 클래스가 다른 클래스에 사용되는 item의 경우 마지막에 item을 붙인다.
- item의 경우 어떤 클래스에서 사용이 되었는지 알 수 있게 한다.



3) 메소드 이름

- 메소드는 Camel Casing 표기법을 따른다. (소문자 시작, 구분되는 단어 대문자)
- 메소드의 이름만 보고 어떠한 기능을 하는지 알 수 있게 한다.

```
private void getCommunityDetail()
private void layoutSetting()
```

- 메소드를 이용하여 다른 타입의 자원을 접근하여 읽어오는 메소드에는 'get'을 붙인다.
- 다른 타입의 값을 접근하여 변경시키는 메소드는 'set'을 붙인다.

```
public String getErr()
public void setCount(String count)
```

4) 변수 이름

- 오브젝트의 변수일 경우 오브젝트 + 변수명을 사용한다.
- 변수는 주로 소문자로 시작하며, 약어를 사용한다.

```
private SimpleSideDrawer slidingMenu;
private TabLayout tabLayout;
private ListView lstNavItem;
private ImageView imgNavUser;
private TextView txtNavUserNickName;
```

- boolean 타입 변수의 이름은 부정적으로 쓰지 않는다.

```
isError (O)
isNoError (X)
```

- 상수는 대문자로 시작하며, 구분은 _를 사용한다.

```
final static public String BASE_URL
final static public String IMAGE_URL
final static public String IMAGE_UPLOAD_URL
```

- 약어를 사용하지 못할 경우 클래스의 앞 글자를 따서 변수의 첫 이름으로 사용한다.

```
public class MainListItem {

    private int mNumber;
    private String mCost;
    private String mTime;
    private String mLike;
    private int mImg;
    private String mUrl;
```

5) 주석 규칙

- 클래스의 최상단은 /** */를 이용하며, 생성한 사람과 생성한 날짜를 명시하고, 클래스의 대한 설명을 작성한다.

```
/**
 * Created by MiSeon on 2016-05-18.
 * 커뮤니티 작성
 */
public class CommunityCreateActivity extends AppCompatActivity {
```

- 메소드의 위에는 /** */를 이용하여 메서드의 기능에 대한 설명을 작성한다.
- 파라미터가 있을 경우 파라미터에 대한 명시를 해준다.

```
/**
 * @param requestCode
 * @param resultCode
 * @param data
 *
 * parameter
 * StringArrayList image
 * StringArrayList explain
 *
 * 액티비티로 돌아왔을 때 이미지와 설명을 추가한다.
 */
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
```

나. 프로그램 목록

1) 전체 프로그램의 이름과 하는 일 간단히 서술

가) 패키지 소개

상위 패키지	하위 패키지	설명
activity	community	커뮤니티와 관련된 액티비티 패키지
	credit	결제와 관련된 액티비티 패키지
	gcm	푸시 알람과 관련된 액티비티 패키지
	intro	시작 화면과 관련된 액티비티 패키지
	knowhow	노하우 작성/조회 등 노하우와 관련된 액티비티 패키지
	material	재료와 관련된 액티비티 패키지
	member	로그인/회원 등 멤버와 관련된 액티비티 패키지
	navigation	네비게이션과 관련된 액티비티 패키지
	search	검색과 관련된 액티비티 패키지
	settings	설정과 관련된 패키지
adapter		리스트, 그리드뷰 등에 사용되는 Adapter 패키지
dao		서버와 연결을 위한 패키지
dto		서버와 연결을 위한 Bean 패키지
fragment		Fragment 패키지
items		리스트, 그리드뷰 등에 사용되는 Item 패키지
service		애플리케이션의 Back단에서 돌아가는 서비스 패키지
utils		기타 필요한 기능들을 위한 패키지
leeform		최상위 패키지

나) 클래스 소개

패키지	클래스	설명
community	CommunityCreateActivity	커뮤니티 생성 액티비티
	CommunityDetailActivity	커뮤니티 상세화면 액티비티
credit	DemoCreditPage	결제 데모 액티비티
gcm	PopUpActivity	푸시 알람을 받았을 때의 화면 액티비티
intro	SplashActivity	시작 화면 액티비티
knowhow	CreateKnowHowExplainActivity	노하우 제작 방법에 대한 상세 설명(사진, 설명 추가)화면 액티비티
	DetailKnowHowActivity	노하우 상세화면 액티비티
	DetailKnowHowReplyActivity	노하우 상세화면의 댓글화면 액티비티
	WriteKnowHowActivity	노하우 작성 액티비티
material	MaterialDetailActivity	재료 상세화면 액티비티
member	LoginActivity	로그인 액티비티
	RegisterActivity	회원가입 액티비티
navigation	MyPageActivity	마이페이지 액티비티
	MypageModifyActivity	마이페이지 수정 액티비티
	PurchaseListActivity	구입 리스트 액티비티

	RegistSellerActivity	판매자 등록 액티비티
	SaleListActivity	판매 리스트 액티비티
search	KnowHowSearchActivity	노하우 검색 액티비티
	MaterialSearchActivity	재료 검색 액티비티
settings	AboutUsActivity	AboutUs 액티비티
	LicenseActivity	라인선스 리스트 액티비티
	LicenseDetailActivity	라인선스 디테일 액티비티
	NoticeActivity	공지사항 액티비티
	SettingActivity	설정 액티비티
adapter	CommunityListAdapter	메인 화면 커뮤니티 리스트 어댑터
	CommunityReplyListAdapter	커뮤니티 댓글 리스트 어댑터
	CreateKnowHowGridAdapter	노하우 제작 방법에 대한 상세 설명 그리드 어댑터
	KakaoSDKAdapter	카카오SDK 어댑터
	LicenseListAdapter	라이선스 리스트 어댑터
	MainFragmentAdapter	메인 탭뷰를 위한 프래그먼트 어댑터
	MainListAdapter	메인 화면 노하우 리스트 어댑터
	MaterialGridAdapter	메인 화면 재료 그리드 어댑터
	MypageFragmentAdapter	마이페이지 탭뷰를 위한 프래그먼트 어댑터
	NoticeListAdapter	공지사항 리스트 어댑터
dao	ConnectService	서버 연결
dto	CommunityDetailBean	커뮤니티 상세화면을 위한 Bean
	CommunityListBean	메인 화면 커뮤니티에 띄워줄 아이템을 위한 Bean
	CommunityWritingBean	커뮤니티 글쓰기를 위한 Bean
	KnowHowBean	메인 화면 노하우 조회를 위한 Bean
	KnowHowDetailBean	노하우 디테일 조회를 위한 Bean
	KnowHowWritingBean	노하우를 작성하기 위한 Bean
	LicenseBean	라이선스 조회를 위한 Bean
	MaterialDetailBean	재료 디테일 조회를 위한 Bean
	MaterialListBean	메인 화면 재료 조회를 위한 Bean
	NoticeBean	공지사항 조회를 위한 Bean
	OnlyErrBean	에러코드만 있는 Bean (insert, update 구문)
	RegistBean	판매자 등록을 위한 Bean
	UserBean	사용자의 정보 조회를 위한 Bean
	UserInfoBean	카카오 유저 정보를 읽고 쓰기 위한 Bean
	UserModifyBean	사용자의 정보 수정을 위한 Bean
	WritingDetailReplyLi	노하우 상세화면의 댓글작성을 위한 Bean

	stBean	
fragment	CommunityFragment	메인 화면에 보여줄 커뮤니티 프래그먼트
	HomeFragment	메인 화면에 보여줄 노하우 프래그먼트
	MaterialFragment	메인 화면에 보여줄 재료 프래그먼트
	MypageScrapFragment	내 정보 화면에 보여줄 스크랩한 노하우 프래그먼트
	MypageWriteFragment	내 정보 화면에 보여줄 내가 작성한 노하우 프래그먼트
items	CommunityDetailBeanItem	CommunityDetailBean에 사용되는 Item
	CommunityItem	Community에 사용되는 Item
	CommunityListBeanItem	CommunityListBean에 사용되는 Item
	CommunityReplyBeanItem	CommunityReplyBean에 사용되는 Item
	CreateKnowHowItem	CreateKnowHow에 사용되는 Item
	KnowHowDetailBeanContentsItem	KnowHowDetail의 사진부분을 제외한 나머지(제목, 난이도, 소요비용 등)에 사용되는 Item
	KnowHowDetailBeanPictureItem	KnowHowDetail의 사진부분에 사용되는 Item
	KnowHowWritingBeanItem	KnowHowWritingBean에 사용되는 Item
	LicenseBeanItem	LicenseBean에 사용되는 Item
	LicenseItem	License 리스트에 사용되는 Item
	MainListItem	노하우 리스트에 사용되는 Item
	MaterialDetailBeanItem	MaterialDetailBean에 사용되는 Item
	MaterialDetailBeanPictureItem	MaterialDetail의 사진부분에 사용되는 Item
	MaterialGridItem	MaterialGrid에 사용되는 Item
	MaterialListBeanItem	MaterialListBean에 사용되는 Item
	NoticeBeanItem	NoticeBean에 사용되는 Item
	NoticeItem	공지사항 리스트에 사용되는 Item
	ReplyItem	Reply에 사용되는 Item
	UserBeanItem	UserBean에 사용되는 Item
	UserInfoBeanItem	UserInfoBean에 사용되는 Item
	WritingBeanItem	WritignBean에 사용되는 Item
	WritingDetailReplyListBeanItem	WritingDetailReplyListBean에 사용되는 Item
service	MyGcmListenerService	서버로부터 푸시 알람이 왔을 시 메시지를 받기 위한 Service
	MyInstanceDListenerService	Token 값을 생성하기 위한 Service 클래스
	RegistrationIntentService	생성된 토큰값을 저장하기 위한 Service

utils	BackPressedHandler	뒤로 가기 이벤트 클래스
	DataProvider	재료 사진 저장 클래스
	DownloadImageTask	이미지 다운로드 클래스
	EndString	글자 수가 어느 정도 이상 넘어가면 ... 으로 말줄임표를 해주는 클래스
	GlobalApplication	카카오 SDK에 필요한 클래스
	LoadActivityList	모든 액티비티 종료를 위한 클래스
	NumZeroForm	재료의 여러 사진을 보여주기 위한 폼 클래스
	QuickstarPreferences	GCM 사용을 위해 기본적으로 세팅해주어야 하는 Preference
	RoundImageView	원형 이미지 뷰 클래스
	SaveData	판매자 권한 체크를 위한 데이터 저장 클래스
	SaveDataMemberInfo	상태 정보 세션 유지를 위한 데이터 저장 클래스
	StaticURL	기본이 되는 URL들이 들어있는 클래스
	TimeMaximum	시간의 최대치를 설정해 둔 클래스
	TimeTransform	현재시간에서 글이 작성된 시간까지 얼마나 걸렸는지 변환해주는 클래스
leeform	MainActivity	가장 기본이 되는 메인 액티비티

다) 기능 별 소개

System	내.꾸.우	Subject	프로그램 수행 역할	작성자	김헌진
				Page	1 / 4
기능	파일명	설명			
내가 꾸미는 우리 집 (내.꾸.우) 메인	MainActivity.java	애플리케이션의 가장 기본이 되는 화면으로, 탭뷰를 이용하여 노하우, 재료, 커뮤니티의 화면을 보여준다. 또한, 네비게이션을 통해 내 정보, 구매목록, 판매자등록(판매내역), 설정이 가능하다.			
	HomeFragment.java	로그인을 하고 가장 먼저 나오는 화면이며 노하우의 리스트를 보여준다. 노하우 리스트에는 노하우 사진, 노하우 제목, 노하우의 간단한 설명 등이 적혀 있으며, 데이터 사용량을 줄이기 위해 한 번에 가져오는 리스트의 개수에 제한을 두었다. Floating Action Button을 사용하여 노하우를 작성화면으로 넘어갈 수 있다.			
	MaterialFragment.java	재료 리스트를 보여주는 화면이다. 사용자는 등록된 재료만 볼 수 있으며, 그리드뷰를 통해 한 줄에 2개씩 보여진다. 그리드뷰에는 재료사진, 이름, 가격이 표시된다.			
	CommunityFragment.java	커뮤니티 리스트를 보여주는 화면이다. 내.꾸.우 사용자들은 이 커뮤니티 기능을 통해 서로 소통을 할 수 있다. 노하우 리스트와 마찬가지로 Floating Action Button을 통해 커뮤니티 작성이 가능하다. 커뮤니티 리스트에는 작성자, 커뮤니티 글, 댓글 수 등을 보여준다.			

System	내.꾸.우	Subject	프로그램 수행 역할	작성자	김현진
				Page	2 / 4
기능	파일명	설명			
노하우	CreateKnowHowExp lainActivity.java	노하우 작성 페이지에서 + 버튼을 클릭했을 때 나오는 화면으로 상세 노하우 작성 페이지이다. 그림을 추가하고 그에 대한 설명을 작성할 수 있다.			
	DetailKnowHowActi vity.java	메인 화면 노하우 리스트에서 리스트를 클릭했을 때 나오는 화면이다. 노하우에 대한 제작 정보, 제작 순서 등을 보여준다. 제작 순서에는 사용자가 작성한 순서대로 설명과 그림을 보여준다. 작성자가 유튜브 코드를 입력했을 경우 유튜브 동영상이 보이게 된다.			
	DetailKnowHowRep lyActivity.java	노하우 상세화면에서 댓글 이미지를 클릭한 경우 나오는 화면이다. 댓글을 작성하고 등록버튼을 누르면 댓글이 등록된다.			
	WriteKnowHowActi vity.java	메인 화면 노하우 리스트에서 FloatingActionButton을 클릭했을 때 나오는 화면이다. 노하우를 작성하는 화면으로 제목, 간단한 설명, 제작 정보 등을 작성하며, +버튼을 통해 노하우의 사진과 설명 추가가 가능하다.			
재료	MaterialDetailActivit y.java	메인 화면 재료 리스트에서 재료를 클릭했을 때 나오는 화면이다. 재료 판매 업체에서 올린 재료들의 사진과 주요 설명, 가격 등이 나와 있다. 사진은 뷰페이저를 통해 가로로 넘기면서 볼 수 있다.			
커뮤니티	CommunityCreateA ctivity.java	메인 화면 커뮤니티 리스트에서 FloatingActionButton을 클릭했을 때 나오는 화면이다. 커뮤니티 작성 기능으로 간단하게 글을 작성할 수 있다.			
	CommunityDetailAc tivity.java	메인 화면 커뮤니티 리스트 중 리스트를 클릭했을 때 나오는 화면이다. 커뮤니티 글의 전체가 보여지며, 커뮤니티에 댓글을 달아 작성자 혹은 비슷한 취미를 가진 사람들과 소통을 할 수 있다.			

System	내.꾸.우	Subject	프로그램 수행 역할	작성자	김헌진
				Page	3 / 4
기능	파일명	설명			
멤버	LoginActivity.java	애플리케이션을 처음 사용하는 경우 나오는 로그인 화면이다. 로그인 카카오톡을 사용하였다. 처음 로그인을 하게 되면 이후에는 자동으로 로그인이 된다.			
	RegisterActivity.java	카카오톡 회원이 아닌 경우 카카오톡 회원 가입 화면으로 이동하게 된다.			
네비게이션	MyPageActivity.java	네비게이션에서 내 정보를 클릭하면 볼 수 있는 화면이다. 사용자의 이미지와 내가 작성한 노하우, 스크랩한 노하우를 조회할 수 있다. 정보수정 버튼을 클릭하면 자신의 개인 정보들을 수정할 수 있다.			
	MyPageModifyActivity.java	정보수정 버튼을 클릭했을 때 나오는 화면이다. 사용자는 닉네임, 주소, 계좌명의, 은행명, 계좌번호, 휴대폰번호를 수정할 수 있다.			
	PurchaseListActivity.java	네비게이션에 있는 화면 중 구매목록을 누르면 볼 수 있는 화면이다. 자신이 구매한 노하우나 재료의 리스트를 볼 수 있다.			
	RegistSellerActivity.java	네비게이션 중 판매자 등록 항목을 클릭하면 볼 수 있는 화면이다. 판매자 등록 항목은 사용자의 권한이 판매자가 아닌 경우 나타난다. 노하우 작성을 위해서는 판매자 등록이 필요하며, 판매자 등록을 위한 정보를 입력하게 된다.			
	SaleListActivity.java	네비게이션 중 판매 내역을 클릭하면 볼 수 있는 화면이다. 판매 내역 항목은 사용자의 권한이 판매자인 경우 나타나며, 판매 내역을 조회할 수 있다.			

System	내.꾸.우	Subject	프로그램 수행 역할	작성자	김헌진
				Page	4 / 4
기능	파일명	설명			
설정	AboutUsActivity.java	AboutUs에 관한 내용으로, 내.꾸.우에 관한 간단한 설명과 만든 사람들이 나온다.			
	LicenseActivity.java	License의 리스트를 볼 수 있는 화면이다.			
	LicenseDetailActivity.java	License 리스트 중 하나를 클릭했을 때 나오는 상세화면이다.			
	NoticeActivity.java	내.꾸.우 관리자가 등록한 공지사항을 확인할 수 있다. 네비게이션의 설정을 통해 들어갈 수 있다. 공지사항은 DB에 저장되어 변경이 유용하다.			
	SettingActivity.java	푸시알람, 회원 탈퇴 등에 관련된 설정을 할 수 있는 화면이다. 네비게이션의 설정을 통해 들어갈 수 있으며, 회원 탈퇴를 하게 될 경우 모든 데이터는 삭제가 된다.			

2) 안드로이드 스튜디오 파일 구조

클래스	.xml
<ul style="list-style-type: none"> ▼ sungkyul.ac.kr.leeform <ul style="list-style-type: none"> ▼ activity <ul style="list-style-type: none"> ▼ community <ul style="list-style-type: none"> CommunityCreateActivity CommunityDetailActivity ▼ credit <ul style="list-style-type: none"> DemoCreditPage ▼ gcm <ul style="list-style-type: none"> PopUpActivity ▼ Intro <ul style="list-style-type: none"> SplashActivity ▼ knowhow <ul style="list-style-type: none"> CreateKnowHowExplainActivity DetailKnowHowActivity DetailKnowHowReplyActivity WriteKnowHowActivity ▼ material <ul style="list-style-type: none"> MaterialDetailActivity ▼ member <ul style="list-style-type: none"> LoginActivity RegisterActivity ▼ navigation <ul style="list-style-type: none"> MyPageActivity MyPageModifyActivity PurchaseListActivity RegistSellerActivity SaleListActivity ▼ search <ul style="list-style-type: none"> KnowHowSearchActivity MaterialSearchActivity ▼ settings <ul style="list-style-type: none"> AboutUsActivity LicenseActivity LicenseDetailActivity NoticeActivity SettingActivity ▼ adapter <ul style="list-style-type: none"> CommunityListAdapter CommunityReplyListAdapter CreateKnowHowGridAdapter KakaoSDKAdapter LicenseListAdapter MainFragmentAdapter MainListAdapter MaterialGridAdapter MypageFragmentAdapter NoticeListAdapter ▼ dao <ul style="list-style-type: none"> ConnectService ▼ dto <ul style="list-style-type: none"> AlarmCheckBean CommunityDetailBean CommunityListBean CommunityWritingBean getAlarmStateBean KnowHowBean KnowHowDetailBean KnowHowWritingBean LicenseBean MaterialDetailBean MaterialListBean NoticeBean OnlyForBean RegistBean UserBean UserInfoBean UserModifyBean WritingDetailReplyListBean ▼ fragment <ul style="list-style-type: none"> CommunityFragment HomeFragment MaterialFragment MypageScrapFragment MypageWriteFragment ▼ Items <ul style="list-style-type: none"> CommunityDetailBeanItem CommunityItem CommunityListBeanItem CommunityReplyBeanItem CreateKnowHowItem getAlarmStateBeanItem KnowHowDetailBeanContentsItem KnowHowDetailBeanPictureItem KnowHowWritingBeanItem LicenseBeanItem LicenseItem MainListItem MaterialDetailBeanItem MaterialDetailBeanPictureItem MaterialGridItem MaterialListBeanItem NoticeBeanItem NoticeItem ReplyItem UserBeanItem UserInfoBeanItem WritingBeanItem WritingDetailReplyListBeanItem ▼ service <ul style="list-style-type: none"> MyGcmListenerService MyInstanceIDListenerService RegistrationIntentService ▼ utils <ul style="list-style-type: none"> BackPressedHandler DataProvider DownloadImageTask EndString GlobalApplication LoadActivityList NumZeroForm QuickstartPreferences RoundImageView SaveData SaveDataMemberInfo StaticURL TimeMaximum TimeTransform MainActivity 	<ul style="list-style-type: none"> ▼ res <ul style="list-style-type: none"> ▼ drawable ▼ layout <ul style="list-style-type: none"> activity_community_create.xml activity_community_detail.xml activity_create_know_how.xml activity_create_know_how_explain.xml activity_create_know_how_view.xml activity_demo_credit_page.xml activity_know_how_detail.xml activity_know_how_detail_create_info.xml activity_know_how_detail_user_info.xml activity_know_how_search.xml activity_knowhow_detail_past.xml activity_konw_how_write.xml activity_login.xml activity_main.xml activity_material_detail.xml activity_material_search.xml activity_mypage.xml activity_mypage_userimage.xml activity_mypagemodify.xml activity_notice.xml activity_pop_up.xml activity_purchase_list.xml activity_pwsearch.xml activity_regist_seller.xml activity_register.xml activity_splash.xml custom_preference_screen_layout.xml fragment_community.xml fragment_home.xml fragment_material.xml fragment_mypage.xml header_community_detail.xml item_grid_create.xml item_grid_material.xml item_knowhow_detail.xml item_list_community.xml item_list_main.xml item_list_reply.xml item_spinner.xml item_tab_layout.xml nav_item.xml nav_view.xml numfield.xml toolbar_custom_form.xml toolbar_custom_mypage.xml toolbar_custom_ok.xml toolbar_custom_search.xml

다. 프로그램 소스(source)

상위패키지	activity	하위패키지	community
CommunityCreateActivity.java			
<pre> package sungkyul.ac.kr.leeform.activity.community; import android.os.Bundle; import android.support.v7.app.AppCompatActivity; import android.support.v7.widget.Toolbar; import android.util.Log; import android.view.View; import android.widget.EditText; import android.widget.ImageView; import android.widget.TextView; import android.widget.Toast; import java.util.HashMap; import java.util.Map; import retrofit2.Call; import retrofit2.Callback; import retrofit2.Response; import retrofit2.Retrofit; import retrofit2.converter.gson.GsonConverterFactory; import sungkyul.ac.kr.leeform.R; import sungkyul.ac.kr.leeform.dao.ConnectService; import sungkyul.ac.kr.leeform.dto.CommunityWritingBean; import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo; import sungkyul.ac.kr.leeform.utils.StaticURL; /** * Created by MiSeon on 2016-05-18. * 커뮤니티 작성 */ public class CommunityCreateActivity extends AppCompatActivity { EditText edtCommunity; Toolbar toolbar; TextView txtToolBarTitle; ImageView imgBack, imgOk; private static String URL = StaticURL.BASE_URL; @Override protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity_community_create); layoutSetting(); toolbar.setContentInsetsAbsolute(0, 0); //툴바 텍스트 변경 txtToolBarTitle.setText("커뮤니티 작성"); //뒤로가기 버튼 imgBack.setOnClickListener(new View.OnClickListener() { </pre>			


```

@Override
public void onClick(View v) {
    finish();
}
});
//확인 버튼
imgOk.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        //내용을 작성하지 않았을 때 메시지 띄우기
        if (edtCommunity.getText().toString().equals("")) {
            Toast.makeText(getApplicationContext(), "내용을 입력해주세요",
            Toast.LENGTH_SHORT).show();
        } else {
            //내용을 작성했을 때
            setCommunityCreate();
            finish();
        }
    }
});
}
/**
 * 레이아웃 셋팅
 */
private void layoutSetting() {
    toolbar = (Toolbar) findViewById(R.id.toolbarBack);
    txtToolBarTitle = (TextView) findViewById(R.id.txtToolBarTitle);
    imgBack = (ImageView) findViewById(R.id.imgBackOk);
    edtCommunity = (EditText) findViewById(R.id.edtCommunity);
    imgOk = (ImageView) findViewById(R.id.imgOk);
}
/**
 * 커뮤니티 작성하기
 * 서버에 커뮤니티 내용과 작성자 키 보내기
 */
private void setCommunityCreate() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    String content = edtCommunity.getText().toString();
    Map<String, String> data = new HashMap<>();
    String userKey = SaveDataMemberInfo.getAppPreferences(getApplicationContext(),
    "user_key");
    data.put("user_unique_key", userKey);
    data.put("community_writing_contents", content);
}

```

```

ConnectService connectService = retrofit.create(ConnectService.class);
Call<CommunityWritingBean> call = connectService.setCommunity(data);
call.enqueue(new Callback<CommunityWritingBean>() {
    @Override
    public void onResponse(Call<CommunityWritingBean> call,
Response<CommunityWritingBean> response) {
        CommunityWritingBean decodedResponse = response.body();
        //Err값이 0이면 성공
        if (decodedResponse.getErr().equals("0")) {
            Toast.makeText(getApplicationContext(), "완료", Toast.LENGTH_SHORT).show();
        }
    }
    @Override
    public void onFailure(Call<CommunityWritingBean> call, Throwable t) {
        Log.e("onFailure", t.getMessage());
    }
});
}
}

```

CommunityDetailActivity.java

```

package sungkyul.ac.kr.leeform.activity.community;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.view.View;
import android.view.inputmethod.InputMethodManager;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import java.text.ParseException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.adapter.CommunityReplyListAdapter;
import sungkyul.ac.kr.leeform.dao.ConnectService;

```

```

import sungkyul.ac.kr.leeform.dto.CommunityDetailBean;
import sungkyul.ac.kr.leeform.dto.CommunityListBean;
import sungkyul.ac.kr.leeform.items.ReplyItem;
import sungkyul.ac.kr.leeform.utils.DownloadImageTask;
import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo;
import sungkyul.ac.kr.leeform.utils.StaticURL;
import sungkyul.ac.kr.leeform.utils.TimeTransForm;
/**
 * Created by MiSeon on 2016-05-18.
 * 커뮤니티 디테일
 * 커뮤니티 리스트에서 선택한 커뮤니티와 댓글을 나타낸다.
 */
public class CommunityDetailActivity extends AppCompatActivity {
    private Toolbar toolbar;
    private CommunityReplyListAdapter adapter;
    ArrayList<ReplyItem> listItem = new ArrayList<>();
    private static String URL = StaticURL.BASE_URL;
    int number;
    TextView content, replyCount, userName, txtReplyRegister, txtTime, txtToolBarTitle;
    ImageView userImg, imgBack;
    EditText edtContents;
    ListView lst;
    View header;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_community_detail);
        toolbar = (Toolbar) findViewById(R.id.toolbarBack);
        toolbar.setContentInsetsAbsolute(0, 0);
        //툴바 완료버튼 보이지 않게 하기
        ImageView imgOk = (ImageView) findViewById(R.id.imgOk);
        imgOk.setVisibility(View.INVISIBLE);
        //뒤로가기 버튼
        imgBack = (ImageView) findViewById(R.id.imgBackOk);
        imgBack.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
        //inflater를 통해 item_list_community.xml을 가져온다.
        header = getLayoutInflater().inflate(R.layout.header_community_detail, null, false);
        //adapter를 통해 item.list.reply.xml을 ArrayList(listItem)에 설정한다.
        adapter = new CommunityReplyListAdapter(getApplicationContext(), R.layout.item_list_reply,
            listItem);
        lst = (ListView) findViewById(R.id.replyList);
    }
}

```

```

//lst 윗부분에 넣기
lst.addHeaderView(header);
//lst에 adapter를 등록한다.
lst.setAdapter(adapter);
layoutSetting();
getCommunityDetail();
getCommunityReply();
//댓글 등록 시
txtReplyRegister.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if (edtContents.getText().toString().isEmpty()) {
            Toast.makeText(getApplicationContext(), "댓글을 입력해주세요.",
Toast.LENGTH_SHORT).show();
            return;
        } else {
            setCommunityReply();
            edtContents.setText("");
            //스크린키보드
            InputMethodManager keyboard = (InputMethodManager)
getSystemService(Context.INPUT_METHOD_SERVICE);
            keyboard.hideSoftInputFromWindow(edtContents.getWindowToken(), 0);
        }
    }
});
}

/**
 * 레이아웃 셋팅
 */
private void layoutSetting() {
    txtTime = (TextView) header.findViewById(R.id.txtHeaderCommunityTime);
    content = (TextView) header.findViewById(R.id.txtHeaderContentCommunity);
    replyCount = (TextView) findViewById(R.id.txtHeaderReplyCount);
    userName = (TextView) header.findViewById(R.id.txtHeaderCommunityUserName);
    userImg = (ImageView) header.findViewById(R.id.imgHeaderCommunityDetail);
    txtReplyRegister = (TextView) findViewById(R.id.txtCommunityReplyRegister);
    edtContents = (EditText) findViewById(R.id.edtCommunityReplyContents);
    txtToolBarTitle = (TextView) findViewById(R.id.txtToolBarTitle);
}

/**
 * 커뮤니티 디테일부분
 * 선택한 커뮤니티의 작성자이름,내용,이미지 가져오는 부분
 * 가져온 내용을 TextView나 ImageView에 설정
 * parameter
 * int Number
 * String name

```

```

* String image
* String time
* String contents
*/
private void getCommunityDetail() {
    Intent it = getIntent();
    number = it.getExtras().getInt("Number");
    Log.e("number", number + "");
    content.setText(it.getExtras().getString("contents"));
    userName.setText(it.getExtras().getString("name"));
    txtTime.setText(it.getExtras().getString("time"));
    new DownloadImageTask(userImg).execute(it.getExtras().getString("image"));
    adapter.notifyDataSetChanged();
}

/**
 * 커뮤니티 댓글부분
 * 댓글 작성한 닉네임,내용,작성자 이미지 가져오기
 * 가져온 내용을 리스트에 추가
 */
private void getCommunityReply() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    final Call<CommunityDetailBean> call = connectService.get(number + "");
    call.enqueue(new Callback<CommunityDetailBean>() {
        @Override
        public void onResponse(Call<CommunityDetailBean> call,
            Response<CommunityDetailBean> response) {
            Log.e("response", response.code() + "");
            Log.e("selectnumber", number + "");
            //CommunityBeanDetail로 디코딩
            CommunityDetailBean decode = response.body();
            Log.e("replycount", decode.getCommunity_reply().size() + "");
            //선택한 커뮤니티의 댓글수를 TextView에 설정
            replyCount.setText(decode.getCommunity_reply().size() + "");
            listItem.clear();
            //listItem에 ReplyItem(댓글작성한닉네임,내용,작성자 이미지를 추가
            for (int i = 0; i < decode.getCommunity_reply().size(); i++) {
                try {
                    listItem.add(new
                        ReplyItem(decode.getCommunity_reply().get(i).getName(),
                            decode.getCommunity_reply().get(i).getReply_community_contents(),
                            decode.getCommunity_reply().get(i).getImg(),
                            TimeTransForm.formatTimeString(decode.getCommunity_reply().get(i).getReply_date())));
                } catch (ParseException e) {

```

```

        e.printStackTrace();
    }
}
adapter.notifyDataSetChanged();
}
@Override
public void onFailure(Call<CommunityDetailBean> call, Throwable t) {
    Log.e("failure", t.getMessage());
}
});
}
/**
 * 커뮤니티 댓글 작성 부분
 * 사용자 유저 키와 댓글 내용, 커뮤니티 키를 서버에 보낸다.
 */
private void setCommunityReply() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    Map<String, String> data = new HashMap<>();
    data.put("user_unique_key",
        SaveDataMemberInfo.getAppPreferences(getApplicationContext(), "user_key"));
    data.put("community_unique_key", number + "");
    data.put("reply_community_contents", edtContents.getText().toString());
    Call<CommunityListBean> call = connectService.setCommunityReply(data);
    call.enqueue(new Callback<CommunityListBean>() {
        @Override
        public void onResponse(Call<CommunityListBean> call, Response<CommunityListBean>
response) {
            //CommunityBeanDetail로 디코딩
            CommunityListBean decode = response.body();
            if (decode.getErr().equals("0")) {
                getCommunityReply();
            }
        }
    });
    @Override
    public void onFailure(Call<CommunityListBean> call, Throwable t) {
        Log.e("failure", t.getMessage());
    }
});
}
}
}

```

상위패키지	activity	하위패키지	credit
DemoCreditPage.java			
<pre> package sungkyul.ac.kr.leeform.activity.credit; import android.os.Bundle; import android.support.v7.app.AppCompatActivity; import android.webkit.WebView; import android.webkit.WebViewClient; import sungkyul.ac.kr.leeform.R; public class DemoCreditPage extends AppCompatActivity { @Override protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity_demo_credit_page); String url = "https://web.nicepay.co.kr/smart/mainPay.jsp"; WebView webView = (WebView) findViewById(R.id.webViewCreditPage); webView.setWebViewClient(new WebViewClient()); // 이걸 안해주면 새창이 뜸 webView.clearCache(true); webView.clearHistory(); webView.getSettings().setJavaScriptEnabled(true); webView.getSettings().setAllowContentAccess(true); webView.getSettings().setBlockNetworkLoads(false); webView.getSettings().setJavaScriptCanOpenWindowsAutomatically(true); webView.getSettings().setAppCacheEnabled(false); webView.loadUrl(url); } } </pre>			

상위패키지	activity	하위패키지	gcm
PopUpActivity.java			
<pre> package sungkyul.ac.kr.leeform.activity.gcm; import android.app.Activity; import android.content.Intent; import android.os.Bundle; import android.view.View; import android.view.WindowManager; import android.widget.Button; import android.widget.TextView; import sungkyul.ac.kr.leeform.R; import sungkyul.ac.kr.leeform.activity.intro.SplashActivity; public class PopUpActivity extends Activity { @Override protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity_pop_up); getWindow().addFlags(WindowManager.LayoutParams.FLAG_SHOW_WHEN_LOCKED // 키잠금 해제하기 </pre>			

```

        | WindowManager.LayoutParams.FLAG_DISMISS_KEYGUARD
        // 화면 켜기
        | WindowManager.LayoutParams.FLAG_TURN_SCREEN_ON);
TextView tv = (TextView) findViewById(R.id.tv_content);
Intent it = getIntent();
String message = it.getExtras().getString("message");
tv.setText(message);
Button btnOk = (Button) findViewById(R.id.btn_ok);
btnOk.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intent = new Intent(PopUpActivity.this, SplashActivity.class);
        startActivity(intent);
        finish();
    }
});
Button btnCancel = (Button) findViewById(R.id.btn_cancel);
btnCancel.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        finish();
    }
});
}
}
}

```

상위패키지	activity	하위패키지	intro
SplashActivity.java			
<pre> package sungkyul.ac.kr.leeform.activity.intro; import android.content.Intent; import android.os.Bundle; import android.support.v7.app.AppCompatActivity; import sungkyul.ac.kr.leeform.R; import sungkyul.ac.kr.leeform.activity.member.LoginActivity; /** * Created by YongHoon on 2016-06-03. * 메인 스플래시 작성 */ public class SplashActivity extends AppCompatActivity { @Override protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity_splash); // 화면이 보이고 2초 뒤에 로그인 페이지로 넘어감 Thread mTimer = new Thread() { @Override </pre>			


```

        public void run() {
            try {
                sleep(2000);
                Intent mIntro = new Intent(SplashActivity.this, LoginActivity.class);
                startActivity(mIntro);
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    };
    mTimer.start();
}
@Override
protected void onPause() {
    super.onPause();
    finish();
}
}

```

상위패키지	activity	하위패키지	knowhow
CreateKnowHowExplainActivity.java			
<pre> package sungkyul.ac.kr.leeform.activity.knowhow; import android.app.AlertDialog; import android.app.Dialog; import android.content.DialogInterface; import android.content.Intent; import android.database.Cursor; import android.graphics.Bitmap; import android.net.Uri; import android.os.Bundle; import android.os.Environment; import android.provider.MediaStore; import android.support.v7.app.AppCompatActivity; import android.support.v7.widget.Toolbar; import android.util.Log; import android.view.View; import android.widget.EditText; import android.widget.ImageView; import android.widget.TextView; import com.squareup.picasso.Picasso; import java.io.BufferedOutputStream; import java.io.File; import java.io.FileOutputStream; import java.util.ArrayList; import sungkyul.ac.kr.leeform.R; </pre>			

```

/**
 * Created by HunJin on 2016-05-19.
 * 노하우의 사진과 설명을 작성
 */
public class CreateKnowHowExplainActivity extends AppCompatActivity {
    private static final int PICK_FROM_CAMERA = 0;
    private static final int PICK_FROM_ALBUM = 1;
    private static final int CROP_FROM_IMAGE = 2;
    private String absolutePath;
    private Uri mImageCpatureUri;
    private String strContents;
    private EditText edtContents;
    ImageView img, imgOk, imgCancel;
    private Toolbar toolbar;
    ArrayList<String> strUrl;
    ArrayList<String> strExplain;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_create_know_how_explain);
        Intent it = getIntent();
        strUrl = it.getStringArrayListExtra("image"); // 이미지 배열 저장을 위한 리스트
        strExplain = it.getStringArrayListExtra("explain"); // 설명 저장을 위한 리스트
        toolbar = (Toolbar) findViewById(R.id.toolbarBack);
        toolbar.setContentInsetsAbsolute(0, 0);
        TextView txtTitle = (TextView) findViewById(R.id.txtToolBarTitle);
        txtTitle.setText("노하우 추가하기");
        imgOk = (ImageView) findViewById(R.id.imgOk);
        imgCancel = (ImageView) findViewById(R.id.imgBackOk);
        imgCancel.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
        edtContents = (EditText) findViewById(R.id.edtCreateExplain);
        // 이미지 클릭 했을 때
        img = (ImageView) findViewById(R.id.imgCreateExplain);
        img.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Dialog.OnClickListner cameraListner = new DialogInterface.OnClickListner() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        doTakePhotoAction();
                    }
                }
            }
        });
    }
}

```

```

    };
    Dialog.OnClickListener albumListener = new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            doTakeAlbumAction();
        }
    };
    Dialog.OnClickListener cancelListener = new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            dialog.dismiss();
        }
    };
    new AlertDialog.Builder(CreateKnowHowExplainActivity.this)
        .setTitle("이미지 선택")
        .setPositiveButton("취소", cancelListener)
        .setNeutralButton("사진촬영", cameraListener)
        .setNegativeButton("앨범선택", albumListener)
        .show();
}

});
// 버튼 클릭 했을 때
imgOk.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        strContents = edtContents.getText().toString();
        if (!absoultePath.equals("") && !strContents.equals("")) {
            strUrl.add(absoultePath);
            strExplain.add(strContents);
            Intent i = new Intent();
            i.putExtra("image", strUrl);
            i.putExtra("explain", strExplain);
            setResult(RESULT_OK, i);
            Log.e("result", RESULT_OK + "");
            finish();
        }
    }
});
}
// 카메라 촬영 후 이미지 가져오기
public void doTakePhotoAction() {
    Intent it = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    String url = "tmp_" + String.valueOf(System.currentTimeMillis()) + ".jpg"; // 임시 경로 생성
    mImageCpatureUri = Uri.fromFile(new File(Environment.getExternalStorageDirectory(), url));
    // 임시 파일 생성
    it.putExtra(MediaStore.EXTRA_OUTPUT, mImageCpatureUri);
}

```

```

        startActivityResult(it, PICK_FROM_CAMERA);
    }
    // 앨범에서 이미지 가져오기
    public void doTakeAlbumAction() {
        Intent it = new Intent(Intent.ACTION_PICK);
        it.setType(MediaStore.Images.Media.CONTENT_TYPE);
        startActivityResult(it, PICK_FROM_ALBUM);
    }
    /**
     * @param requestCode
     * @param resultCode
     * @param data
     */
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if (resultCode != RESULT_OK) {
            return;
        }
        // 받아온 값이 있으면
        switch (requestCode) {
            // 앨범에서 가져오는 경우 카메라에서 가져오는 것과 같은 기능을 하기에 break 없이
            진행
            case PICK_FROM_ALBUM: {
                mImageCpatureUri = data.getData();
            }
            // 카메라에서 가져올 경우
            case PICK_FROM_CAMERA: {
                // 이미지 크롭
                absolutePath = getPath(mImageCpatureUri);
                Picasso.with(getApplicationContext()).load(mImageCpatureUri).resize(930,
                    501).centerCrop().into(img);
                Log.w("absolutePath", absolutePath);
                break;
            }
            case CROP_FROM_IMAGE: {
                if (resultCode != RESULT_OK) {
                    return;
                }
                final Bundle extras = data.getExtras();
                // 경로에 이미지 저장 (임시 파일)
                String filePath = Environment.getExternalStorageDirectory().getAbsolutePath() +
                    "/LeeForm/" + System.currentTimeMillis() + ".jpg";
                if (extras != null) {
                    Log.e("extras", extras + "");
                    Bitmap photo = extras.getParcelable("data"); // 크롭한 이미지를 가져옴

```

```

        Log.e("photo", photo + "");
        img.setImageBitmap(photo); // 이미지 로드
        storeCropImage(photo, filePath); // 크롭한 이미지 저장, 안하게 되면 이미지를
가져오지 못함
        absoltePath = filePath; // 경로 반환
        break;
    }
    File f = new File(mImageCpatureUri.getPath()); // 임시 파일 삭제
    if (f.exists()) {
        f.delete();
    }
}
}
private String getPath(Uri uri) {
    String[] projection = {MediaStore.Images.Media.DATA};
    Cursor cursor = managedQuery(uri, projection, null, null, null);
    startManagingCursor(cursor);
    int columnIndex = cursor.getColumnIndexOrThrow(MediaStore.Images.Media.DATA);
    cursor.moveToFirst();
    return cursor.getString(columnIndex);
}
// 크롭한 이미지 저장
private void storeCropImage(Bitmap bitmap, String filePath) {
    String dirPath = Environment.getExternalStorageDirectory().getAbsolutePath() + "/LeeForm";
    File directory_LeeForm = new File(dirPath);
    if (!directory_LeeForm.exists()) {
        directory_LeeForm.mkdir();
    }
    File copyFile = new File(filePath);
    BufferedOutputStream out = null;
    try {
        copyFile.createNewFile();
        out = new BufferedOutputStream(new FileOutputStream(copyFile));
        bitmap.compress(Bitmap.CompressFormat.JPEG, 100, out);
        sendBroadcast(new Intent(Intent.ACTION_MEDIA_SCANNER_SCAN_FILE,
Uri.fromFile(copyFile)));
        out.flush();
        out.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
}

```

DetailKnowHowActivity.java

package sungkyul.ac.kr.leeform.activity.knowhow;

```

import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.view.View;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.TextView;
import com.squareup.picasso.Picasso;
import java.util.HashMap;
import java.util.Map;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.activity.credit.DemoCreditPage;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.KnowHowDetailBean;
import sungkyul.ac.kr.leeform.dto.OnlyErrBean;
import sungkyul.ac.kr.leeform.dto.WritingDetailReplyListBean;
import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo;
import sungkyul.ac.kr.leeform.utils.StaticURL;

/**
 * Created by YongHoon on 2016-05-18.
 * 노하우 상세정보
 */
public class DetailKnowHowActivity extends AppCompatActivity {
    StringBuffer sbContent;
    View layKnowhowDetail;
    private ImageView imgScrapImage;
    private LinearLayout btnKnowHowDetailShare, btnKnowHowDetailReply,
    btnKnowHowDetailScrap;
    private Toolbar toolbar;
    private ImageView imgKnowHowDetailMain, imgKnowHowDetailUserInfo,
    imgKnowHowDetailBuying;
    private TextView txtKnowHowDetailName, txtKnowHowDetailShortExplain,
    txtKnowHowDetailTime, txtKnowHowDetailUserName;
    private TextView txtKnowHowDetailLevel, txtKnowHowDetailMakeTime,
    txtKnowHowDetailMakingPrice, txtToolBarTitle;
    private TextView txtKnowHowDetailReplyCount, txtKnowHowDetailScrapCount;
    private boolean isScrap = false;
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {

```

```

super.onCreate(savedInstanceState);
setContentView(R.layout.activity_know_how_detail);
// 공유할때 보낼 컨텐츠 담을 버퍼 생성
sbContent = new StringBuffer();
//툴바 완료버튼 보이지 않게 하기
ImageView imgOk = (ImageView) findViewById(R.id.imgOk);
imgOk.setVisibility(View.INVISIBLE);
//뒤로가기 버튼
ImageView imgBack = (ImageView) findViewById(R.id.imgBackOk);
imgBack.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        finish();
    }
});
initializeLayout();
imgKnowHowDetailBuying.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        startActivity(new Intent(getApplicationContext(), DemoCreditPage.class));
    }
});
btnKnowHowDetailShare.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent itSendContent = new Intent();
        itSendContent.setAction(Intent.ACTION_SEND);
        itSendContent.putExtra(Intent.EXTRA_TEXT, sbContent.toString());
        itSendContent.setType("text/plain");
        startActivity(itSendContent);
    }
});
Intent it = getIntent();
final String knowHowKey = it.getExtras().getString("knowhowkey");
String imageUrl = it.getExtras().getString("image");
Picasso.with(getApplicationContext()).load(imageUrl).resize(1080,
720).centerCrop().into(imgKnowHowDetailMain);
getItem(knowHowKey);
getKnowHowReplyCount(knowHowKey);
scrapCheck(knowHowKey);
btnKnowHowDetailReply.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent it = new Intent(getApplicationContext(), DetailKnowHowReplyActivity.class);
        it.putExtra("key", knowHowKey);
        startActivity(it);
    }
});

```

```

        overridePendingTransition(R.anim.common_slide_bottom_to_up,
R.anim.common_slide_bottom_to_up);
    }

});

btnKnowHowDetailScrap.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        // 스크랩 처리
        if(isScrap == true) {
            // 스크랩 취소
            cancelScrap(knowHowKey);
        } else {
            // 스크랩
            setScrap(knowHowKey);
        }
        scrapCheck(knowHowKey);
    }
});
}

/**
 * 설명과 사진(동영상) 아이템을 넣는 메소드
 *
 * @param position 해당 아이템이 몇번째인지
 * @param explain 해당 아이템의 설명
 * @param url 사진이나 동영상의 url
 */
private void makeKnowhowItem(int position, String explain, String url) {
    View itemView;
    itemView = View.inflate(getApplicationContext(), R.layout.item_knowhow_detail, null);
    ImageView imageView = (ImageView)
itemView.findViewById(R.id.imgKnowHowDetailContentsPicture);
    Picasso.with(getApplicationContext()).load(url).resize(1030,
600).centerCrop().into(imageView);
    TextView tv1 = (TextView) itemView.findViewById(R.id.tvKnowhowDetailExplain);
    tv1.setText(position + ". " + explain);
    LinearLayout layKnowhowDetailItem = (LinearLayout)
itemView.findViewById(R.id.layKnowhowDetailItem);
    ((LinearLayout) layKnowhowDetail).addView(layKnowhowDetailItem);
}

/**
 * 화면에 보여줄 정보 초기화
 */
private void initializeLayout() {
    btnKnowHowDetailShare = (LinearLayout)findViewById(R.id.btnKnowHowDetailShare);
    layKnowhowDetail = findViewById(R.id.layKnowhowDetail);
    toolbar = (Toolbar) findViewById(R.id.toolbarBack);
    toolbar.setContentInsetsAbsolute(0, 0);
}

```



```

imgKnowHowDetailBuying = (ImageView) findViewById(R.id.imgKnowHowDetailBuying);
imgKnowHowDetailMain = (ImageView) findViewById(R.id.imgKnowHowDetailMain);
imgKnowHowDetailUserInfo = (ImageView) findViewById(R.id.imgKnowHowDetailUserInfo);
txtKnowHowDetailLevel = (TextView) findViewById(R.id.txtKnowHowDetailLevel);
txtKnowHowDetailMakeTime = (TextView) findViewById(R.id.txtKnowHowDetailMakeTime);
txtKnowHowDetailMakingPrice = (TextView) findViewById(R.id.txtKnowHowDetailMakingPrice);
txtKnowHowDetailName = (TextView) findViewById(R.id.txtKnowHowDetailName);
txtKnowHowDetailShortExplain = (TextView) findViewById(R.id.txtKnowHowDetailShortExplain);
txtKnowHowDetailUserName = (TextView) findViewById(R.id.txtKnowHowDetailUserName);
txtKnowHowDetailTime = (TextView) findViewById(R.id.txtKnowHowDetailTime);
txtToolBarTitle = (TextView) findViewById(R.id.txtToolBarTitle);
btnKnowHowDetailReply = (LinearLayout) findViewById(R.id.btnKnowHowReply);
btnKnowHowDetailScrap = (LinearLayout) findViewById(R.id.btnKnowHowScrap);
txtKnowHowDetailScrapCount = (TextView) findViewById(R.id.txtKnowHowDetailScrapCount);
txtKnowHowDetailReplyCount = (TextView) findViewById(R.id.txtKnowHowDetailReplyCount);
imgScrapImage = (ImageView) findViewById(R.id.imgScrapImage);
}

private void getItem(String writingUniqueKey) {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    Log.e("writingKey", writingUniqueKey);
    Call<KnowHowDetailBean> call = connectService.getKnowHowDetail(writingUniqueKey);
    call.enqueue(new Callback<KnowHowDetailBean>() {
        @Override
        public void onResponse(Call<KnowHowDetailBean> call,
            Response<KnowHowDetailBean> response) {
            KnowHowDetailBean decode = response.body();
            Log.e("size", decode.getWriting_data1().get(0).getImg() + "");

            Picasso.with(getApplicationContext()).load(decode.getWriting_data1().get(0).getImg()).resize(0,
                imgKnowHowDetailUserInfo.getHeight()).into(imgKnowHowDetailUserInfo);

            txtKnowHowDetailLevel.setText(decode.getWriting_data1().get(0).getLevel().toString());

            txtKnowHowDetailMakeTime.setText(decode.getWriting_data1().get(0).getMaking_time().toString());

            txtKnowHowDetailMakingPrice.setText(decode.getWriting_data1().get(0).getCost().toString());

            txtKnowHowDetailName.setText(decode.getWriting_data1().get(0).getWriting_name().toString());

            txtKnowHowDetailShortExplain.setText(decode.getWriting_data1().get(0).getExplanation().toString());
        }
    });
}

```

```

txtKnowHowDetailUserName.setText(decode.getWriting_data1().get(0).getName().toString());

txtKnowHowDetailTime.setText(decode.getWriting_data1().get(0).getWriting_date());
txtToolBarTitle.setText(decode.getWriting_data1().get(0).getWriting_name());

txtKnowHowDetailScrapCount.setText(decode.getWriting_data1().get(0).getScrap_amount().toString());

sbContent.append("★"
decode.getWriting_data1().get(0).getWriting_name().toString() + "★\n" +
    decode.getWriting_data1().get(0).getExplanation().toString() +
    "\n\n" +
    "난이도 : " + decode.getWriting_data1().get(0).getLevel().toString() + " / "
+
    "소요시간 : " + decode.getWriting_data1().get(0).getCost().toString() + " / "
" +
    "소요비용 : " + decode.getWriting_data1().get(0).getCost().toString() +
    "\n\n" +
    "<제작방법>" +
    "\n"
);
for (int i = 0; i < decode.getWriting_data2().size(); i++) {
    makeKnowhowItem(i
decode.getWriting_data2().get(i).getWriting_contents().toString(),
decode.getWriting_data2().get(i).getPicture_url().toString());
    sbContent.append((i+1)
decode.getWriting_data2().get(i).getWriting_contents().toString());
    if(i+1 != decode.getWriting_data2().size()){
        sbContent.append("\n");
    }
}
}
@Override
public void onFailure(Call<KnowHowDetailBean> call, Throwable t) {
}
});
}
private void getKnowHowReplyCount(String writingUniqueKey) {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    Call<WritingDetailReplyListBean> call
connectService.getWritingDetailReply(writingUniqueKey);
    call.enqueue(new Callback<WritingDetailReplyListBean>() {
        @Override
        public void onResponse(Call<WritingDetailReplyListBean> call,
Response<WritingDetailReplyListBean> response) {
            WritingDetailReplyListBean decode = response.body();

```

```

        txtKnowHowDetailReplyCount.setText(decode.getReply_list().size()+"");
    }
    @Override
    public void onFailure(Call<WritingDetailReplyListBean> call, Throwable t) {
    }
    });
}
private void scrapCheck(final String writingKey) {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();
    ConnectService connectService = retrofit.create(ConnectService.class);
    Map<String, String> map = new HashMap<>();
    map.put("user_unique_key", SaveDataMemberInfo.getAppPreferences(getApplicationContext(),"user_key"));
    map.put("writing_unique_key",writingKey);
    Call<OnlyErrBean> call = connectService.getCheckScrap(map);
    call.enqueue(new Callback<OnlyErrBean>() {
        @Override
        public void onResponse(Call<OnlyErrBean> call, Response<OnlyErrBean> response) {
            OnlyErrBean decode = response.body();
            if(decode.getErr()=="3") {
                isScrap = true;
            }
            imgScrapImage.setImageDrawable(getResources().getDrawable(R.drawable.knowhow_like));
        } else if(decode.getErr()=="5") {
            isScrap = false;
        }
        imgScrapImage.setImageDrawable(getResources().getDrawable(R.drawable.knowhow_unlike));
    }
    getKnowHowScrapCount(writingKey);
}
    @Override
    public void onFailure(Call<OnlyErrBean> call, Throwable t) {
    }
    });
}
private void cancelScrap(String writingKey) {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();
    ConnectService connectService = retrofit.create(ConnectService.class);
    Map<String, String> map = new HashMap<>();
    map.put("user_unique_key",SaveDataMemberInfo.getAppPreferences(getApplicationContext(),"use

```

```

r_key"));
    map.put("writing_unique_key",writingKey);
    Call<OnlyErrBean> call = connectService.unScrap(map);
    call.enqueue(new Callback<OnlyErrBean>() {
        @Override
        public void onResponse(Call<OnlyErrBean> call, Response<OnlyErrBean> response) {
        }
        @Override
        public void onFailure(Call<OnlyErrBean> call, Throwable t) {
        }
    });
}

private void setScrap(String writingKey) {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    Map<String, String> map = new HashMap<>();

    map.put("user_unique_key",SaveDataMemberInfo.getAppPreferences(getApplicationContext(),"user_key"));
    map.put("writing_unique_key",writingKey);
    Call<OnlyErrBean> call = connectService.setScrap(map);
    call.enqueue(new Callback<OnlyErrBean>() {
        @Override
        public void onResponse(Call<OnlyErrBean> call, Response<OnlyErrBean> response) {
        }
        @Override
        public void onFailure(Call<OnlyErrBean> call, Throwable t) {
        }
    });
}

private void getKnowHowScrapCount(String writingUniqueKey) {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    Log.e("writingKey", writingUniqueKey);
    Call<KnowHowDetailBean> call = connectService.getKnowHowDetail(writingUniqueKey);
    call.enqueue(new Callback<KnowHowDetailBean>() {
        @Override
        public void onResponse(Call<KnowHowDetailBean> call,
        Response<KnowHowDetailBean> response) {
            KnowHowDetailBean decode = response.body();

```

```

txtKnowHowDetailScrapCount.setText(decode.getWriting_data1().get(0).getScrap_amount().toString(0));
    }
    @Override
    public void onFailure(Call<KnowHowDetailBean> call, Throwable t) {
    }
    });
}
}
}

```

DetailKnowHowReplyActivity.java

```

package sungkyul.ac.kr.leeform.activity.knowhow;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.inputmethod.InputMethodManager;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.adapter.CommunityReplyListAdapter;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.CommunityDetailBean;
import sungkyul.ac.kr.leeform.dto.CommunityListBean;
import sungkyul.ac.kr.leeform.dto.OnlyErrBean;
import sungkyul.ac.kr.leeform.dto.WritingDetailReplyListBean;
import sungkyul.ac.kr.leeform.items.ReplyItem;
import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo;
import sungkyul.ac.kr.leeform.utils.StaticURL;
public class DetailKnowHowReplyActivity extends Activity {
    EditText edtContents;
    TextView txtRegist;
    ListView lstReply;
}

```

```

private InputMethodManager keyboard;
private CommunityReplyListAdapter adapter;
ArrayList<ReplyItem> listItem = new ArrayList<>();
int number;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_detail_know_how_reply);
    Intent it = getIntent();
    number = Integer.parseInt(it.getExtras().getString("key"));
    layoutSetting();
    lstReply.setAdapter(adapter);
    Log.e("numer", number + "");
    getWritingReply();
    txtRegist.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if (edtContents.getText().toString().isEmpty()) {
                Toast.makeText(getApplicationContext(), "댓글을 입력해주세요.",
                Toast.LENGTH_SHORT).show();
                return;
            } else {
                setKnowHowDetailReply();
                edtContents.setText("");
                //스크린키보드
                InputMethodManager keyboard = (InputMethodManager)
getSystemService(Context.INPUT_METHOD_SERVICE);
                keyboard.hideSoftInputFromWindow(edtContents.getWindowToken(), 0);
            }
        }
    });
}

private void layoutSetting() {
    edtContents = (EditText) findViewById(R.id.edtKnowHowDetailReplyContents);
    txtRegist = (TextView) findViewById(R.id.txtKnowHowDetailReplyRegister);
    adapter = new CommunityReplyListAdapter(getApplicationContext(), R.layout.item_list_reply,
listItem);
    lstReply = (ListView) findViewById(R.id.lstKnowHowDetailReply);
    keyboard = (InputMethodManager) getSystemService(Context.INPUT_METHOD_SERVICE);
    keyboard.hideSoftInputFromWindow(edtContents.getWindowToken(), 0);
    listItem.clear();
}

private void setKnowHowDetailReply() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())

```

```

        .build();

        ConnectService connectService = retrofit.create(ConnectService.class);
        Map<String, String> data = new HashMap<>();
        data.put("user_unique_key",
SaveDataMemberInfo.getAppPreferences(getApplicationContext(), "user_key"));
        data.put("writing_unique_key", number + "");
        data.put("reply_writing_contents", edtContents.getText().toString());
        Call<OnlyErrBean> call = connectService.setWritingDetailReply(data);
        call.enqueue(new Callback<OnlyErrBean>() {
            @Override
            public void onResponse(Call<OnlyErrBean> call, Response<OnlyErrBean> response) {
                getWritingReply();
            }
            @Override
            public void onFailure(Call<OnlyErrBean> call, Throwable t) {
            }
        });
    }

    private void getWritingReply() {
        Retrofit retrofit = new Retrofit.Builder()
            .baseUrl(StaticURL.BASE_URL)
            .addConverterFactory(GsonConverterFactory.create())
            .build();

        ConnectService connectService = retrofit.create(ConnectService.class);
        Call<WritingDetailReplyListBean> call = connectService.getWritingDetailReply(number + "");
        call.enqueue(new Callback<WritingDetailReplyListBean>() {
            @Override
            public void onResponse(Call<WritingDetailReplyListBean> call,
Response<WritingDetailReplyListBean> response) {
                WritingDetailReplyListBean decode = response.body();
                int size = decode.getReply_list().size();
                Log.e("size", size + "");
                listItem.clear();
                for (int i = 0; i < size; i++) {
                    listItem.add(new ReplyItem(decode.getReply_list().get(i).getName().toString(),
                        decode.getReply_list().get(i).getReply_writing_contents(),
                        decode.getReply_list().get(i).getImg(),
                        decode.getReply_list().get(i).getReply_date()));
                }
                adapter.notifyDataSetChanged();
            }
            @Override
            public void onFailure(Call<WritingDetailReplyListBean> call, Throwable t) {
            }
        });
    }
}

```

```

}
WriteKnowHowActivity.java
package sungkyul.ac.kr.leeform.activity.knowhow;
import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.GridView;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.adapter.CreateKnowHowGridAdapter;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.KnowHowWritingBean;
import sungkyul.ac.kr.leeform.items.CreateKnowHowItem;
import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo;
import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * Created by user on 2016-06-08.
 */
public class WriteKnowHowActivity extends AppCompatActivity {
    String upLoadServerUri = StaticURL.IMAGE_UPLOAD_URL; // 파일 업로드를 위한 php url 이다.
    String URL = StaticURL.BASE_URL; // api 기본 베이스 주소

```



```

String imageUrl = StaticURL.IMAGE_URL; // 이미지 저장 위치
int serverResponseCode = 0;
String uploadFilePath; // 파일 경로
String uploadFileName; // 파일 이름
String writingUniqueKey;
LinearLayout linearSell;
String level;
String check_sale;
String selectedCategory;
String selectedMakingTime;
Spinner spnCategory, spnMakingTime;
EditText edtName, edtCost, edtSellAmount, edtSellPrice, edtYoutuCode, edtExplanation;
LinearLayout lineCreateView;
ImageView imgOk, imgBack;
TextView tvTitle;
Toolbar toolbar;
Button btnSellYes, btnSellNo, btnLevelHigh, btnLevelMiddle, btnLevelLow;
String[] category;
String[] makingTime;
private GridView grvCreate;
private CreateKnowHowGridAdapter cAdapter;
private ArrayList<String> strUrl = new ArrayList<>(); //파일 경로리스트
private ArrayList<String> strExplain = new ArrayList<>(); //파일 설명 리스트트
ArrayList<CreateKnowHowItem> gridItems = new ArrayList<>();
@Override
protected void onCreate(@Nullable Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_konw_how_write);
    layoutSetting();
    toolbar = (Toolbar) findViewById(R.id.toolbarBack);
    toolbar.setContentInsetsAbsolute(0, 0);
    tvTitle.setText("노하우 작성");
    //기본값
    btnSellNo.setSelected(true);
    btnLevelHigh.setSelected(true);
    ArrayAdapter<String> spinnerCategoryAdapter = new
ArrayAdapter<String>(getApplicationContext(), R.layout.item_spinner, category);
    ArrayAdapter<String> spinnerMakingTimeAdapter = new
ArrayAdapter<String>(getApplicationContext(), R.layout.item_spinner, makingTime);

    spinnerCategoryAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_
item);

    spinnerMakingTimeAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_
item);

    spnCategory.setAdapter(spinnerCategoryAdapter); //스피너에 adapter 설정
    spnMakingTime.setAdapter(spinnerMakingTimeAdapter);

```

```

//노하우 카테고리 선택 시
spnCategory.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
    @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
        String str = (String) spnCategory.getSelectedItem().toString();
        selectedCategory = str;
    }
    @Override
    public void onNothingSelected(AdapterView<?> parent) {
    }
});

//제작 시간 선택 시
spnMakingTime.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
    @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
        String str = (String) spnMakingTime.getSelectedItem().toString();
        selectedMakingTime = str;
    }
    @Override
    public void onNothingSelected(AdapterView<?> parent) {
    }
});

cAdapter = new CreateKnowHowGridAdapter(getApplicationContext(),
R.layout.item_grid_create, gridItems);
grvCreate.setAdapter(cAdapter);
check_sale = "0"; //판매NO
level = btnLevelHigh.getText().toString(); //레벨 상
init();
imgBack.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        finish();
    }
});
imgOk.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        check();
        if (check() == false) {
            writeKnowHow();
            finish();
        }
    }
});

// + 이미지 클릭 시
grvCreate.setOnItemClick(new AdapterView.OnItemClickListener() {

```

```

@Override
public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
    if (position == gridItems.size() - 1) {
        Intent intent = new Intent(getApplicationContext(),
CreateKnowHowExplainActivity.class);
        intent.putExtra("image", strUrl);
        intent.putExtra("explain", strExplain);
        startActivityForResult(intent, 1000);
    }
}
});
//판매 Yes 버튼 클릭시
btnSellYes.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        btnSellYes.setSelected(true);
        btnSellNo.setSelected(false);
        linearSell.setVisibility(View.VISIBLE);
        check_sale = "1";
    }
});
//판매 No 버튼 클릭시
btnSellNo.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        btnSellYes.setSelected(false);
        btnSellNo.setSelected(true);
        linearSell.setVisibility(View.INVISIBLE);
        check_sale = "0";
    }
});
//난이도 상 버튼 클릭시
btnLevelHigh.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        btnLevelHigh.setSelected(true);
        btnLevelMiddle.setSelected(false);
        btnLevelLow.setSelected(false);
        level = btnLevelHigh.getText().toString();
    }
});
//난이도 중 버튼 클릭시
btnLevelMiddle.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        btnLevelHigh.setSelected(false);

```

```

        btnLevelMiddle.setSelected(true);
        btnLevelLow.setSelected(false);
        level = btnLevelMiddle.getText().toString();
    }
});
//난이도 하 버튼 클릭시
btnLevelLow.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        btnLevelHigh.setSelected(false);
        btnLevelMiddle.setSelected(false);
        btnLevelLow.setSelected(true);
        level = btnLevelLow.getText().toString();
    }
});
}
/**
 * 반드시 입력해야하는 값들이
 * 널값인지 체크
 *
 * @return
 */
private boolean check() {
    if (edtName.getText().toString().equals("")) {
        Toast.makeText(getApplicationContext(), "제목을 입력해주세요",
        Toast.LENGTH_SHORT).show();
        return true;
    }
    if (edtExplanation.getText().toString().equals("")) {
        Toast.makeText(getApplicationContext(), "설명을 입력해주세요",
        Toast.LENGTH_SHORT).show();
        return true;
    }
    if (edtCost.getText().toString().equals("")) {
        Toast.makeText(getApplicationContext(), "소요비용을 입력해주세요",
        Toast.LENGTH_SHORT).show();
        return true;
    }
    if (check_sale.equals("1")) {
        if (edtSellAmount.getText().toString().equals("")) {
            Toast.makeText(getApplicationContext(), "판매수량을 입력해주세요",
            Toast.LENGTH_SHORT).show();
            return true;
        }
        if (edtSellPrice.getText().toString().equals("")) {
            Toast.makeText(getApplicationContext(), "판매 가격을 입력해주세요",
            Toast.LENGTH_SHORT).show();

```

```

        return true;
    }
}

if (spnCategory.getSelectedItemId().toString().equals("카테고리")) {
    Toast.makeText(getApplicationContext(), "категори를 선택해주세요",
    Toast.LENGTH_SHORT).show();
    return true;
}

if (spnMakingTime.getSelectedItemId().toString().equals("소요시간")) {
    Toast.makeText(getApplicationContext(), "소요시간을 선택해주세요",
    Toast.LENGTH_SHORT).show();
    return true;
}

return false;
}

/**
 * 레이아웃 셋팅
 */
private void layoutSetting() {
    tvTitle = (TextView) findViewById(R.id.txtToolBarTitle);
    imgOk = (ImageView) findViewById(R.id.imgOk);
    imgBack = (ImageView) findViewById(R.id.imgBackOk);
    edtName = (EditText) findViewById(R.id.edtName);
    lineCreateView = (LinearLayout) findViewById(R.id.lineImgAdd);
    grvCreate = (GridView) findViewById(R.id.grdImgView);
    spnCategory = (Spinner) findViewById(R.id.spnKnowCategory);
    spnMakingTime = (Spinner) findViewById(R.id.spnMakingTime);
    category = getResources().getStringArray(R.array.category);
    makingTime = getResources().getStringArray(R.array.makingTime);
    linearSell = (LinearLayout) findViewById(R.id.linearSell);
    edtCost = (EditText) findViewById(R.id.edtCost);
    edtYoutuCode = (EditText) findViewById(R.id.edtYoutuCode);
    btnSellYes = (Button) findViewById(R.id.btnSellYes);
    btnSellNo = (Button) findViewById(R.id.btnSellNo);
    btnLevelHigh = (Button) findViewById(R.id.btnLevelHigh);
    btnLevelMiddle = (Button) findViewById(R.id.btnLevelMiddle);
    btnLevelLow = (Button) findViewById(R.id.btnLevelLow);
    edtSellAmount = (EditText) findViewById(R.id.edtSellAmount);
    edtSellPrice = (EditText) findViewById(R.id.edtSellPrice);
    edtExplanation = (EditText) findViewById(R.id.edtExplanation);
}

/**
 * 초기화
 */
private void init() {
    gridItems.clear();

```

```

        for (int i = 0; i < strUrl.size(); i++) {
            gridItems.add(new CreateKnowHowItem(i, strUrl.get(i), strExplain.get(i)));
        }
        //사진 추가한 후 +이미지가 뜨도록 +이미지를 GridView에 추가
        gridItems.add(new CreateKnowHowItem(gridItems.size(), R.drawable.plus, "클릭"));
        cAdapter.notifyDataSetChanged();
    }
    /**
     * 기본값 설정
     * 소요비용, 유튜브코드, 판매량, 판매가격이 널값인 경우
     * 0으로 설정
     */
    private void setting() {
        if (edtCost.getText().toString().equals("")) {
            edtCost.setText("0");
        }
        if (edtYoutubuCode.getText().toString().equals("")) {
            edtYoutubuCode.setText("0");
        }
        if (edtSellAmount.getText().toString().equals("")) {
            edtSellAmount.setText("0");
        }
        if (edtSellPrice.getText().toString().equals("")) {
            edtSellPrice.setText("0");
        }
    }
    /**
     * 노하우 작성시 입력한 값들을 서버에 보내기
     * <p>
     * parameter
     * user_unique_key
     * category_name
     * writing_name
     * check_sales
     * video_url
     * cost
     * making_time
     * price
     * level
     * writing_explanation
     * amount
     */
    private void writeKnowHow() {
        Retrofit retrofit = new Retrofit.Builder()
            .baseUrl(URL)
            .addConverterFactory(GsonConverterFactory.create())

```

```

        .build();

        ConnectService connectService = retrofit.create(ConnectService.class);
        String userUniqueKey = SaveDataMemberInfo.getAppPreferences(getApplicationContext(),
"user_key");
        Map<String, String> data = new HashMap<>();
        setting();
        data.put("user_unique_key", userUniqueKey);
        data.put("category_name", selectedCategory);
        data.put("writing_name", edtName.getText().toString());
        data.put("check_video", "1"); // 변경 필요
        data.put("check_sales", check_sale);
        data.put("video_url", edtYoutuCode.getText().toString());
        data.put("cost", edtCost.getText().toString()); //소요비용
        data.put("making_time", selectedMakingTime);
        data.put("level", level);
        data.put("writing_explanation", "수고가 많습니다.");
        data.put("amount", edtSellAmount.getText().toString());
        data.put("price", edtSellPrice.getText().toString());
        data.put("writing_explanation", edtExplanation.getText().toString());
        Call<KnowHowWritingBean> call = connectService.setKnowGetKey(data);
        call.enqueue(new Callback<KnowHowWritingBean>() {
            @Override
            public void onResponse(Call<KnowHowWritingBean> call,
Response<KnowHowWritingBean> response) {
                KnowHowWritingBean decode = response.body();
                writingUniqueKey = decode.getWriting_unique_key().get(0).getWriting_unique_key();
                for (int i = 0; i < strUrl.size(); i++) { // 생성한 노하우의 수만큼 반복
                    File file = new File(strUrl.get(i)); // 파일을 만들
                    final String contents = strExplain.get(i); // 설명 추출
                    uploadFileName = file.getName(); // 파일의 이름 추출
                    uploadFilePath = file.getPath(); // 파일의 경로 추출
                    new Thread(new Runnable() {
                        @Override
                        public void run() {
                            uploadFile(uploadFilePath); // 파일 업로드
                        }
                    }).start();
                    // 파일이미지, 설명, 우선순위, writing_unique_key
                    leeformParsing(uploadFileName, contents, (i + 1), writingUniqueKey); //
데이터베이스에 저장
                }
            }
            @Override
            public void onFailure(Call<KnowHowWritingBean> call, Throwable t) {
                Log.e("Failure Writing Knowhow", t.getMessage());
            }
        });

```

```

    });
}
/**
 * @param requestCode
 * @param resultCode
 * @param data      parameter
 *                  StringArrayList image
 *                  StringArrayList explain
 *                  액티비티로 돌아왔을 때 이미지와 설명을 추가한다.
 */
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == 1000) {
        if (resultCode == RESULT_OK) {
            //URL과 노하우 설명 가져오기
            strUrl = data.getStringArrayListExtra("image");
            strExplain = data.getStringArrayListExtra("explain");
            init();
        }
    }
}
/**
 * 이미지 올리는 메서드
 *
 * @param sourceFileUri
 * @return
 */
public int uploadFile(String sourceFileUri) {
    String fileName = sourceFileUri;
    HttpURLConnection conn = null;
    DataOutputStream dos = null;
    String lineEnd = "\r\n";
    String twoHyphens = "--";
    String boundary = "*****";
    int bytesRead, bytesAvailable, bufferSize;
    byte[] buffer;
    int maxBufferSize = 10 * 1024 * 1024;
    File sourceFile = new File(sourceFileUri);
    if (!sourceFile.isFile()) {
        Log.e("uploadFile", "Source File not exist ." + sourceFileUri);
        return 0;
    } else {
        try {
            // StaticURL 생성
            FileInputStream fileInputStream = new FileInputStream(sourceFile);

```



```

java.net.URL url = new java.net.URL(uploadServerUri);
// HTTP를 열고 StaticURL 연결
conn = (HttpURLConnection) url.openConnection();
conn.setDoInput(true); // Allow Inputs
conn.setDoOutput(true); // Allow Outputs
conn.setUseCaches(false); // Don't use a Cached Copy
conn.setRequestMethod("POST");
conn.setRequestProperty("Connection", "Keep-Alive");
conn.setRequestProperty("ENCTYPE", "multipart/form-data");
conn.setRequestProperty("Content-Type", "multipart/form-data;boundary=" +
boundary);

conn.setRequestProperty("uploaded_file", fileName);
dos = new DataOutputStream(conn.getOutputStream());
dos.writeBytes(twoHyphens + boundary + lineEnd);
dos.writeBytes("Content-Disposition: form-data; name=\"uploaded_file\";filename=\""
+ fileName + "\"" + lineEnd);
dos.writeBytes(lineEnd);
// 버퍼 생성
bytesAvailable = fileInputStream.available();
bufferSize = Math.min(bytesAvailable, maxBufferSize);
buffer = new byte[bufferSize];
// 버퍼를 사용하여 파일을 읽음
bytesRead = fileInputStream.read(buffer, 0, bufferSize);
while (bytesRead > 0) {
    dos.write(buffer, 0, bufferSize);
    bytesAvailable = fileInputStream.available();
    bufferSize = Math.min(bytesAvailable, maxBufferSize);
    bytesRead = fileInputStream.read(buffer, 0, bufferSize);
}
// multipart 방식으로 파일 전송
dos.writeBytes(lineEnd);
dos.writeBytes(twoHyphens + boundary + twoHyphens + lineEnd);
// 서버로부터 받은 response 값
serverResponseCode = conn.getResponseCode();
String serverResponseMessage = conn.getResponseMessage();
Log.i("uploadFile", "HTTP Response is : "
+ serverResponseMessage + ": " + serverResponseCode);
if (serverResponseCode == 200) {
    runOnUiThread(new Runnable() {
        public void run() {
            Toast.makeText(getApplicationContext(), "File Upload Complete.",
            Toast.LENGTH_SHORT).show();
        }
    });
}
//close the streams //

```

```

        fileInputStream.close();
        dos.flush();
        dos.close();
    } catch (MalformedURLException ex) {
        ex.printStackTrace();
        runOnUiThread(new Runnable() {
            public void run() {
                Toast.makeText(getApplicationContext(), "MalformedURLException",
                    Toast.LENGTH_SHORT).show();
            }
        });
        Log.e("Upload file to server", "error: " + ex.getMessage(), ex);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return serverResponseCode;
} // End else block
}
@Override
protected void onResume() {
    super.onResume();
}
private void leeformParsing(String fileName, String contents, int priority, String writing_unique_key)
{
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    Log.e("imgUrl", imageStorageUrl + fileName);
    Log.e("contents", contents);
    Log.e("priority", priority + "");
    Log.e("writing_unique_key", writing_unique_key);
    // get 방식 파라미터 전송
    Map<String, String> data = new HashMap<>();
    data.put("imgUrl", imageStorageUrl + fileName);
    data.put("contents", contents);
    data.put("priority", priority + "");
    data.put("writing_unique_key", writing_unique_key);
    data.put("time", System.currentTimeMillis() + "");
    ConnectService connectService = retrofit.create(ConnectService.class);
    Call<KnowHowWritingBean> call = connectService.setKnowHow(data);
    call.enqueue(new Callback<KnowHowWritingBean>() {
        @Override
        public void onResponse(Call<KnowHowWritingBean> call,
            Response<KnowHowWritingBean> response) {
            KnowHowWritingBean des = response.body();

```

```

        Log.e("err", des.getErr());
    }
    @Override
    public void onFailure(Call<KnowHowWritingBean> call, Throwable t) {
        Log.e("failure", t.getMessage());
    }
    });
}
}
}

```

상위패키지	activity	하위패키지	material
MaterialDetailActivity.java			
<pre> package sungkyul.ac.kr.leeform.activity.material; import android.annotation.SuppressLint; import android.annotation.TargetApi; import android.content.Intent; import android.os.Build; import android.os.Bundle; import android.support.annotation.Nullable; import android.support.v4.view.animation.LinearOutSlowInInterpolator; import android.support.v7.app.AppCompatActivity; import android.support.v7.widget.Toolbar; import android.util.Log; import android.view.View; import android.widget.Button; import android.widget.ImageView; import android.widget.TextView; import com.hkm.slider.Animations.DescriptionAnimation; import com.hkm.slider.SliderLayout; import com.hkm.slider.SliderTypes.BaseSliderView; import com.hkm.slider.SliderTypes.TextSliderView; import com.hkm.slider.TransformerL; import com.hkm.slider.Tricks.ViewPagerEx; import com.squareup.picasso.Picasso; import java.util.HashMap; import retrofit2.Call; import retrofit2.Callback; import retrofit2.Response; import retrofit2.Retrofit; import retrofit2.converter.gson.GsonConverterFactory; import sungkyul.ac.kr.leeform.R; import sungkyul.ac.kr.leeform.activity.credit.DemoCreditPage; import sungkyul.ac.kr.leeform.dao.ConnectService; import sungkyul.ac.kr.leeform.dto.MaterialDetailBean; import sungkyul.ac.kr.leeform.utils.DataProvider; import sungkyul.ac.kr.leeform.utils.NumZeroForm; </pre>			

```

import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * Created by KyungHee on 2016-05-19.
 */
public class MaterialDetailActivity extends AppCompatActivity implements
BaseSliderView.OnSliderClickListener, ViewPagerEx.OnPageChangeListener {
    private Toolbar toolbar;
    private SliderLayout mDemoSlider;
    DataProvider dataProvider;
    private Button btnBuying;
    String material_unique_key;
    private ImageView imgMaterialDetailSeller;
    private TextView txtMaterialDetailName, txtMaterialDetailExplain, txtMaterialDetailCost,
    txtToolBarTitle;
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_material_detail);
        Intent it = getIntent();
        material_unique_key = it.getExtras().getString("material_unique_key");
        //툴바 완료버튼 보이지 않게 하기
        ImageView imgOk = (ImageView) findViewById(R.id.imgOk);
        imgOk.setVisibility(View.INVISIBLE);
        //뒤로가기 버튼
        ImageView imgBack = (ImageView) findViewById(R.id.imgBackOk);
        imgBack.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
        initializeLayout();
        getDetailInfo();
        btnBuying.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(getApplicationContext(), DemoCreditPage.class));
            }
        });
    }
    /**
     * 화면에 보여줄 정보 초기화
     */
    private void initializeLayout() {
        toolbar = (Toolbar) findViewById(R.id.toolbarBack);
        toolbar.setContentInsetsAbsolute(0, 0);
    }
}

```

```

        mDemoSlider = (SliderLayout) findViewById(R.id.slider);
        dataProvider = new DataProvider();
        btnBuying = (Button) findViewById(R.id.btnMaterialDetailBuying);
        txtMaterialDetailName = (TextView) findViewById(R.id.txtMaterialDetailName);
        txtMaterialDetailCost = (TextView) findViewById(R.id.txtMaterialDetailCost);
        txtMaterialDetailExplain = (TextView) findViewById(R.id.txtMaterialDetailExplain);
        txtToolBarTitle = (TextView) findViewById(R.id.txtToolBarTitle);
        imgMaterialDetailSeller = (ImageView) findViewById(R.id.imgMaterialDetailSeller);
    }

    // 이미지를 URL로 가져올 땐 이 부분을 사용합니다.
    // 이미지를 src로 가져오기 위해서는 이 부분을 주석처리 하면 됩니다.
    @TargetApi(Build.VERSION_CODES.HONEYCOMB)
    protected void defaultCompleteSlider(final HashMap<String, String> maps) {
        for (String name : maps.keySet()) {
            TextSliderView textSliderView = new TextSliderView(this);
            // initialize a SliderLayout
            textSliderView
                .image(maps.get(name))
                .setScaleType(BaseSliderView.ScaleType.Fit)
                .enableSaveImageByLongClick(getFragmentManager())
                .setOnSliderClickListener(this);

            //add your extra information
            textSliderView.getBundle().putString("extra", name);
            mDemoSlider.addSlider(textSliderView);
        }
    }

    /**
     * 슬라이드를 통해 이미지를 처리하는 매서드입니다.
     */
    @SuppressWarnings("ResourceAsColor")
    private void setupSlider() {
        // remember setup first
        Log.e("setup", "setup slider");
        mDemoSlider.setPresetTransformer(TransformerL.DepthPage);
        mDemoSlider.setPresetIndicator(SliderLayout.PresetIndicators.Center_Bottom);
        mDemoSlider.setCustomAnimation(new DescriptionAnimation());
        mDemoSlider.addOnPageChangeListener(this);
        mDemoSlider.setOffscreenPageLimit(4);
        mDemoSlider.setSliderTransformDuration(400, new LinearOutSlowInInterpolator());
        mDemoSlider.getPagerIndicator().setDefaultIndicatorColor(R.color.colorAccentBasic,
            R.color.colorAccent);

        final NumZeroForm n = new NumZeroForm(this);
        mDemoSlider.setNumLayout(n);
        mDemoSlider.setPresetTransformer("DepthPage");
        mDemoSlider.stopAutoCycle();
        defaultCompleteSlider(dataProvider.getFileSrcHorizontal());
    }

```

```

    }
    @Override
    public void onPageScrolled(int position, float positionOffset, int positionOffsetPixels) {
    }
    @Override
    public void onPageSelected(int position) {
    }
    @Override
    public void onPageScrollStateChanged(int state) {
    }
    @Override
    public void onSliderClick(BaseSliderView coreSlider) {
    }
    @Override
    protected void onStop() {
        super.onStop();
        mDemoSlider.stopAutoCycle();
    }
    /**
     * 재료의 디테일 정보를 가져온다.
     * 재료이름, 재료가격, 재료설명, 재료 이미지
     */
    private void getDetailInfo() {
        Retrofit retrofit = new Retrofit.Builder()
            .baseUrl(StaticURL.BASE_URL)
            .addConverterFactory(GsonConverterFactory.create())
            .build();

        ConnectService connectService = retrofit.create(ConnectService.class);
        Call<MaterialDetailBean> call = connectService.getMaterialDetail(material_unique_key);
        call.enqueue(new Callback<MaterialDetailBean>() {
            @Override
            public void onResponse(Call<MaterialDetailBean> call, Response<MaterialDetailBean>
response) {
                MaterialDetailBean decode = response.body();

                txtMaterialDetailName.setText(decode.getMaterial_detail1().get(0).getMaterial_name().toString());

                txtToolBarTitle.setText(decode.getMaterial_detail1().get(0).getMaterial_name().toString());

                txtMaterialDetailCost.setText(decode.getMaterial_detail1().get(0).getMaterial_price());

                txtMaterialDetailExplain.setText(decode.getMaterial_detail1().get(0).getMaterial_explanation());

                Picasso.with(getApplicationContext()).load(decode.getMaterial_detail1().get(0).getSubcontractor_i
mage_url()).resize(100, 45).centerCrop().into(imgMaterialDetailSeller);

                for (int i = 0; i < decode.getMaterial_detail2().size(); i++) {
                    Log.e("file url",
decode.getMaterial_detail2().get(i).getMaterial_picture_url().toString());
                }
            }
        });
    }

```

```

        dataProvider.setHashFile((i + 1) + "",
decode.getMaterial_detail2().get(i).getMaterial_picture_url().toString());
    }
    setupSlider(); // 이미지 슬라이드를 불러옵니다.
}
@Override
public void onFailure(Call<MaterialDetailBean> call, Throwable t) {
}
});
}
}
}

```

상위패키지	activity	하위패키지	member
LoginActivity.java			
<pre> package sungkyul.ac.kr.leeform.activity.member; import android.content.Context; import android.content.Intent; import android.content.pm.PackageInfo; import android.content.pm.PackageManager; import android.content.pm.Signature; import android.os.Bundle; import android.support.v7.app.AppCompatActivity; import android.util.Log; import android.widget.Toast; import com.kakao.auth.ISessionCallback; import com.kakao.auth.Session; import com.kakao.util.exception.KakaoException; import com.kakao.util.helper.log.Logger; import java.security.MessageDigest; import java.security.NoSuchAlgorithmException; import sungkyul.ac.kr.leeform.R; import sungkyul.ac.kr.leeform.utils.BackPressCloseHandler; import static com.kakao.util.helper.Utility.getPackageInfo; /** * Created by HunJin on 2016-05-14. * 로그인 (카카오) */ public class LoginActivity extends AppCompatActivity { private SessionCallback callback; private BackPressCloseHandler backPressCloseHandler; @Override protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity_login); // 취소버튼 눌렀을 때 핸들러 backPressCloseHandler = new BackPressCloseHandler(this); </pre>			

```

        Log.e("Session", Session.getCurrentSession() + "");
        Log.e("hash", getKeyHash(getApplicationContext()) + "");
        Log.e("session use", Session.getCurrentSession().isOpenable() + "");
        if (Session.getCurrentSession().implicitOpen()) {
            callback = new SessionCallback();
            Session.getCurrentSession().addCallback(callback);
            callback.onSessionOpened();
        } else {
            callback = new SessionCallback();
            Session.getCurrentSession().addCallback(callback);
        }
    }

    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        if (Session.getCurrentSession().handleActivityResult(requestCode, resultCode, data)) {
            return;
        }
        super.onActivityResult(requestCode, resultCode, data);
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        Session.getCurrentSession().removeCallback(callback);
    }

    private class SessionCallback implements ISessionCallback {
        @Override
        public void onSessionOpened() {
            redirectSignupActivity(); // 세션 연결성공 시 redirectSignupActivity() 호출
        }

        @Override
        public void onSessionOpenFailed(KakaoException exception) {
            if (exception != null) {
                Logger.e(exception);
            }
            setContentView(R.layout.activity_login); // 세션 연결이 실패했을때
                                                    // 로그인화면을 다시 불러옴
        }
    }

    protected void redirectSignupActivity() { //세션 연결 성공 시 SignupActivity로 넘김
        final Intent intent = new Intent(this, RegisterActivity.class);
        intent.setFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
        startActivity(intent);
        finish();
    }

    // HashKey 값을 알아보는 메서드 입니다.
    public static String getKeyHash(final Context context) {
        PackageInfo packageInfo = getPackageInfo(context, PackageManager.GET_SIGNATURES);
    }

```



```

        if (packageInfo == null)
            return null;
        for (Signature signature : packageInfo.signatures) {
            try {
                MessageDigest md = MessageDigest.getInstance("SHA");
                md.update(signature.toByteArray());
                return android.util.Base64.encodeToString(md.digest(),
                    android.util.Base64.NO_WRAP);
            } catch (NoSuchAlgorithmException e) {
                Log.w("error", "Unable to getCommunityList MessageDigest. signature=" + signature,
                    e);
            }
        }
        return null;
    }

    /**
     * 뒤로가기 키를 눌렀을 때
     */
    @Override
    public void onBackPressed() {
        //핸들러 작동
        backPressCloseHandler.onBackPressed();
        Toast.makeText(getApplicationContext(), "한 번 더 누르면 앱이 종료됩니다",
            Toast.LENGTH_SHORT).show();
    }
}

```

RegisterActivity.java

```

package sungkyul.ac.kr.leeform.activity.member;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import com.kakao.auth.ErrorCode;
import com.kakao.network.ErrorResult;
import com.kakao.usermgmt.UserManagement;
import com.kakao.usermgmt.callback.MeResponseCallback;
import com.kakao.usermgmt.response.model.UserProfile;
import com.kakao.util.helper.log.Logger;
import sungkyul.ac.kr.leeform.MainActivity;

/**
 * Created by HunJin on 2016-05-14.
 * 로그인 - 회원가입 (카카오)
 */
public class RegisterActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
    }
}

```

```

        requestMe();
    }

    /**
     * 사용자의 상태를 알아 보기 위해 me API 호출을 한다.
     */
    protected void requestMe() { //유저의 정보를 받아오는 함수
        UserManagement.requestMe(new MeResponseCallback() {
            @Override
            public void onFailure(ErrorResult errorResult) {
                Log.e("Failure", errorResult + "");
                String message = "failed to getCommunityList user info. msg=" + errorResult;
                Logger.d(message);
                ErrorCode result = ErrorCode.valueOf(errorResult.getErrorCode());
                if (result == ErrorCode.CLIENT_ERROR_CODE) {
                    finish();
                } else {
                    redirectLoginActivity();
                }
            }
            @Override
            public void onSessionClosed(ErrorResult errorResult) {
                redirectLoginActivity();
            }
            @Override
            public void onNotSignedUp() {
            } // 카카오톡 회원이 아닐 시 showSignup(); 호출해야함
            @Override
            public void onSuccess(UserProfile userProfile) { //성공 시 userProfile 형태로 반환
                Logger.d("UserProfile : " + userProfile);
                Log.e("Success", userProfile + "");
                Log.e("Userid", userProfile.getId() + "");
                redirectMainActivity(userProfile.getId(), userProfile.getNickname(),
                userProfile.getThumbnailImagePath()); // 로그인 성공시 MainActivity로
            }
        });
    }

    // 로그인 성공시
    private void redirectMainActivity(long userId, String nickName, String imagePath) {
        Intent it = new Intent(this, MainActivity.class);
        it.putExtra("UserId", userId);
        it.putExtra("NickName", nickName);
        it.putExtra("Image", imagePath);
        startActivity(it);
        overridePendingTransition(0, 0);
        finish();
    }
}

```

```

protected void redirectLoginActivity() {
    final Intent intent = new Intent(this, LoginActivity.class);
    intent.setFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION);
    startActivity(intent);
    overridePendingTransition(0, 0);
    finish();
}
}

```

상위패키지	activity	하위패키지	navigation
MyPageActivity.java			
<pre> package sungkyul.ac.kr.leeform.activity.navigation; import android.content.Intent; import android.os.Bundle; import android.support.annotation.Nullable; import android.support.design.widget.TabLayout; import android.support.v4.view.ViewPager; import android.support.v7.app.AppCompatActivity; import android.support.v7.widget.Toolbar; import android.util.Log; import android.view.View; import android.widget.ImageView; import android.widget.TextView; import retrofit2.Call; import retrofit2.Callback; import retrofit2.Response; import retrofit2.Retrofit; import retrofit2.converter.gson.GsonConverterFactory; import sungkyul.ac.kr.leeform.R; import sungkyul.ac.kr.leeform.adapter.MypageFragmentAdapter; import sungkyul.ac.kr.leeform.dao.ConnectService; import sungkyul.ac.kr.leeform.dto.UserBean; import sungkyul.ac.kr.leeform.utils.DownloadImageTask; import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo; import sungkyul.ac.kr.leeform.utils.StaticURL; /** * Created by KyungHee on 2016-05-21. */ public class MyPageActivity extends AppCompatActivity { private TabLayout tabLayout; private TextView txtModify, txtToolBarTitle; private ImageView imgBack, imgMypageUser; private Toolbar toolbar; private static String URL = StaticURL.BASE_URL; String imgUrl, image; @Override </pre>			

```

protected void onCreate(@Nullable Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_mypage);
    toolbar = (Toolbar) findViewById(R.id.toolbar);
    toolbar.setContentInsetsAbsolute(0, 0);
    txtToolBarTitle = (TextView) findViewById(R.id.txtToolBarTitle);
    txtToolBarTitle.setText("마이페이지");
    //뒤로가기 버튼
    imgBack = (ImageView) findViewById(R.id.imgBackOk);
    imgBack.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            finish();
        }
    });
    imgMypageUser = (ImageView) findViewById(R.id.imgMypageUser);
    getUserImage();
    txtModify = (TextView) findViewById(R.id.txtMypageModify);
    txtModify.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new Intent(getApplicationContext(), MyPageModifyActivity.class);
            intent.putExtra("imageurl", image);
            startActivity(intent);
        }
    });
    tabInitialization();
}

/**
 * 탭 초기화
 */
private void tabInitialization() {
    ViewPager viewPager = (ViewPager) findViewById(R.id.mainViewPager);
    MypageFragmentAdapter mainFragmentAdapter = new
MypageFragmentAdapter(getSupportFragmentManager(), MyPageActivity.this);
    viewPager.setAdapter(mainFragmentAdapter);
    tabLayout = (TabLayout) findViewById(R.id.myPageTab);
    tabLayout.setupWithViewPager(viewPager);
}

/**
 * 유저 이미지 가져오기
 */
public void getUserImage() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())

```

```

        .build();
        ConnectService connectService = retrofit.create(ConnectService.class);
        String key = SaveDataMemberInfo.getAppPreferences(getApplicationContext(), "user_key");
        final Call<UserBean> call = connectService.getUserDetail(key);
        call.enqueue(new Callback<UserBean>() {
            @Override
            public void onResponse(Call<UserBean> call, Response<UserBean> response) {
                UserBean decodedResponse = response.body();
                imgUrl = decodedResponse.getMyinfo_detail().get(0).getImg();
                new DownloadImageTask(imgMypageUser).execute(imgUrl);
            }
            @Override
            public void onFailure(Call<UserBean> call, Throwable t) {
                Log.e("failure", t.getMessage());
            }
        });
    }
}

```

MypageModifyActivity.java

```

package sungkyul.ac.kr.leeform.activity.navigation;
import android.app.AlertDialog;
import android.app.Dialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.database.Cursor;
import android.graphics.Bitmap;
import android.net.Uri;
import android.os.Bundle;
import android.os.Environment;
import android.provider.MediaStore;
import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;
import com.squareup.picasso.Picasso;
import java.io.BufferedOutputStream;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.net.HttpURLConnection;

```

```

import java.net.MalformedURLException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.UserBean;
import sungkyul.ac.kr.leeform.dto.UserModifyBean;
import sungkyul.ac.kr.leeform.utils.DownloadImageTask;
import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo;
import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * Created by user on 2016-06-10.
 */
public class MyPageModifyActivity extends AppCompatActivity {
    String uploadServerUri = StaticURL.IMAGE_UPLOAD_URL; // 파일 업로드를 위한 php url 이다.
    String imageStorageUrl = StaticURL.IMAGE_URL; // 이미지 저장 위치
    int serverResponseCode = 0;
    EditText edtUserName, edtAddress, edtBankName, edtBankNumber, edtPhoneNumber,
    edtAccountName;
    ImageView image, imgOk, imgBack;
    String URL = StaticURL.BASE_URL;
    private Toolbar toolbar;
    private TextView txtToolbarTitle;
    ;
    private static final int PICK_FROM_CAMERA = 0;
    private static final int PICK_FROM_ALBUM = 1;
    private static final int CROP_FROM_IMAGE = 2;
    private Uri mImageCpatureUri;
    private String absoultePath;
    String uploadFilePath; // 파일 경로
    String uploadFileName; // 파일 이름
    private ArrayList<String> strUrl = new ArrayList<>(); //파일 경로리스트
    private ArrayList<String> strExplain = new ArrayList<>(); //파일 설명 리스트트
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_mypagemodify);
        layoutSetting();
        getUserDetails(); //값가져오기
        toolbar.setContentInsetsAbsolute(0, 0);

```

```

txtToolbarTitle.setText("프로필 수정");
imgBack.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        finish();
    }
});
imgOk.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        {
            if (absoultePath == null) {
                setUserDetails(uploadFileName);
                finish();
            } else {
                File file = new File(absoultePath); // 파일을 만듦
                uploadFileName = file.getName(); // 파일의 이름 추출
                uploadFilePath = file.getPath(); // 파일의 경로 추출
                new Thread(new Runnable() {
                    @Override
                    public void run() {
                        uploadFile(uploadFilePath); // 파일 업로드
                    }
                }).start();
                setUserDetails(uploadFileName);
                finish();
            }
        }
    }
});
image.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Dialog.OnClickListener cameraListener = new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                doTakePhotoAction();
            }
        };
        Dialog.OnClickListener albumListener = new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                doTakeAlbumAction();
            }
        };
        Dialog.OnClickListener cancelListener = new DialogInterface.OnClickListener() {

```

```

        @Override
        public void onClick(DialogInterface dialog, int which) {
            dialog.dismiss();
        }
    };

    new AlertDialog.Builder(MyPageModifyActivity.this)
        .setTitle("이미지 선택")
        .setPositiveButton("취소", cancelListener)
        .setNeutralButton("사진촬영", cameraListener)
        .setNegativeButton("앨범선택", albumListener)
        .show();
    }
});
}

/**
 * 레이아웃 셋팅
 */
private void layoutSetting() {
    edtUserName = (EditText) findViewById(R.id.edtUserName);
    edtAddress = (EditText) findViewById(R.id.edtAddress);
    edtAccountName = (EditText) findViewById(R.id.edtAccountName);
    edtBankName = (EditText) findViewById(R.id.edtBankName);
    edtBankNumber = (EditText) findViewById(R.id.edtBankNumber);
    edtPhoneNumber = (EditText) findViewById(R.id.edtPhoneNumber);
    imgOk = (ImageView) findViewById(R.id.imgOk);
    image = (ImageView) findViewById(R.id.imgMypageUser);
    imgBack = (ImageView) findViewById(R.id.imgBackOk);
    toolbar = (Toolbar) findViewById(R.id.toolbarBack);
    txtToolBarTitle = (TextView) findViewById(R.id.txtToolBarTitle);
}

// 카메라 촬영 후 이미지 가져오기
public void doTakePhotoAction() {
    Intent it = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    String url = "tmp_" + String.valueOf(System.currentTimeMillis()) + ".jpg"; // 임시 경로 생성
    mImageCpatureUri = Uri.fromFile(new File(Environment.getExternalStorageDirectory(), url));
// 임시 파일 생성
    it.putExtra(MediaStore.EXTRA_OUTPUT, mImageCpatureUri);
    startActivityForResult(it, PICK_FROM_CAMERA);
}

// 앨범에서 이미지 가져오기
public void doTakeAlbumAction() {
    Intent it = new Intent(Intent.ACTION_PICK);
    it.setType(MediaStore.Images.Media.CONTENT_TYPE);
    startActivityForResult(it, PICK_FROM_ALBUM);
}

/**

```



```

    * @param requestCode
    * @param resultCode
    * @param data
    */
    @Override
    protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if (resultCode != RESULT_OK) {
            strUrl = data.getStringArrayListExtra("image");
            strExplain = data.getStringArrayListExtra("explain");
            return;
        }
        // 받아온 값이 있으면
        switch (requestCode) {
            // 앨범에서 가져오는 경우 카메라에서 가져오는 것과 같은 기능을 하기에 break 없이
            진행
            case PICK_FROM_ALBUM: {
                mImageCpatureUri = data.getData();
            }
            // 카메라에서 가져올 경우
            case PICK_FROM_CAMERA: {
                // 이미지 크롭
                absoltePath = getPath(mImageCpatureUri);
                Picasso.with(getApplicationContext()).load(mImageCpatureUri).resize(340,
                260).centerCrop().into(image);
                Log.e("absoltePath", absoltePath);
                break;
            }
            case CROP_FROM_IMAGE: {
                if (resultCode != RESULT_OK) {
                    return;
                }
                final Bundle extras = data.getExtras();
                // 경로에 이미지 저장 (임시 파일)
                String filePath = Environment.getExternalStorageDirectory().getAbsolutePath() +
                "/LeeForm/" + System.currentTimeMillis() + ".jpg";
                if (extras != null) {
                    Log.e("extras", extras + "");
                    Bitmap photo = extras.getParcelable("data"); // 크롭한 이미지를 가져옴
                    Log.e("photo", photo + "");
                    image.setImageBitmap(photo); // 이미지 로드
                    storeCropImage(photo, filePath); // 크롭한 이미지 저장, 안하게 되면 이미지를
                    가져오지 못함
                    absoltePath = filePath; // 경로 반환
                    break;
                }
                File f = new File(mImageCpatureUri.getPath()); // 임시 파일 삭제

```

```

        if (f.exists()) {
            f.delete();
        }
    }
}

// 크롭한 이미지 저장
private void storeCropImage(Bitmap bitmap, String filePath) {
    String dirPath = Environment.getExternalStorageDirectory().getAbsolutePath() + "/LeeForm";
    File directory_LeeForm = new File(dirPath);
    if (!directory_LeeForm.exists()) {
        directory_LeeForm.mkdir();
    }
    File copyFile = new File(filePath);
    BufferedOutputStream out = null;
    try {
        copyFile.createNewFile();
        out = new BufferedOutputStream(new FileOutputStream(copyFile));
        bitmap.compress(Bitmap.CompressFormat.JPEG, 100, out);
        sendBroadcast(new Intent(Intent.ACTION_MEDIA_SCANNER_SCAN_FILE,
Uri.fromFile(copyFile)));
        out.flush();
        out.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
}

private String getPath(Uri uri) {
    String[] projection = {MediaStore.Images.Media.DATA};
    Cursor cursor = managedQuery(uri, projection, null, null, null);
    startManagingCursor(cursor);
    int columnIndex = cursor.getColumnIndexOrThrow(MediaStore.Images.Media.DATA);
    cursor.moveToFirst();
    return cursor.getString(columnIndex);
}

/**
 * 사용자의 정보를 가져와서
 * ImageView, EditText에 뿌려주기
 */
private void getUserDetails() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    String key = SaveDataMemberInfo.getAppPreferences(getApplicationContext(), "user_key");

```

```

        final Call<UserBean> call = connectService.getUserDetail(key);
        call.enqueue(new Callback<UserBean>() {
            @Override
            public void onResponse(Call<UserBean> call, Response<UserBean> response) {
                UserBean decodedResponse = response.body();
                String accountNumber =
                decodedResponse.getMyinfo_detail().get(0).getAccount_number();
                String address = decodedResponse.getMyinfo_detail().get(0).getAddress();
                String bankName = decodedResponse.getMyinfo_detail().get(0).getBank_name();
                String name = decodedResponse.getMyinfo_detail().get(0).getName();
                String phoneNumber =
                decodedResponse.getMyinfo_detail().get(0).getPhone_number();
                String accountName =
                decodedResponse.getMyinfo_detail().get(0).getAccount_name();
                edtUserName.setText(name);
                edtAddress.setText(address);
                edtBankName.setText(bankName);
                edtBankNumber.setText(accountNumber);
                edtPhoneNumber.setText(phoneNumber);
                edtAccountName.setText(accountName);

                new
                DownloadImageTask(image).execute(decodedResponse.getMyinfo_detail().get(0).getImg());
            }
            @Override
            public void onFailure(Call<UserBean> call, Throwable t) {
                Log.e("failure", t.getMessage());
            }
        });
    }

    public int uploadFile(String sourceFileUri) {
        String fileName = sourceFileUri;
        HttpURLConnection conn = null;
        DataOutputStream dos = null;
        String lineEnd = "\r\n";
        String twoHyphens = "--";
        String boundary = "*****";
        int bytesRead, bytesAvailable, bufferSize;
        byte[] buffer;
        int maxBufferSize = 10 * 1024 * 1024;
        File sourceFile = new File(sourceFileUri);
        if (!sourceFile.isFile()) {
            Log.e("uploadFile", "Source File not exist : " + sourceFileUri);
            return 0;
        } else {
            try {
                // StaticURL 생성
                FileInputStream fileInputStream = new FileInputStream(sourceFile);

```

```

java.net.URL url = new java.net.URL(uploadServerUri);
// HTTP를 열고 StaticURL 연결
conn = (HttpURLConnection) url.openConnection();
conn.setDoInput(true); // Allow Inputs
conn.setDoOutput(true); // Allow Outputs
conn.setUseCaches(false); // Don't use a Cached Copy
conn.setRequestMethod("POST");
conn.setRequestProperty("Connection", "Keep-Alive");
conn.setRequestProperty("ENCTYPE", "multipart/form-data");
conn.setRequestProperty("Content-Type", "multipart/form-data;boundary=" +
boundary);

conn.setRequestProperty("uploaded_file", fileName);
dos = new DataOutputStream(conn.getOutputStream());
dos.writeBytes(twoHyphens + boundary + lineEnd);
dos.writeBytes("Content-Disposition: form-data;
name=" + uploaded_file + ";filename=" +
+ fileName + ".txt" + lineEnd);
dos.writeBytes(lineEnd);
// 버퍼 생성
bytesAvailable = fileInputStream.available();
bufferSize = Math.min(bytesAvailable, maxBufferSize);
buffer = new byte[bufferSize];
// 버퍼를 사용하여 파일을 읽음
bytesRead = fileInputStream.read(buffer, 0, bufferSize);
while (bytesRead > 0) {
    dos.write(buffer, 0, bufferSize);
    bytesAvailable = fileInputStream.available();
    bufferSize = Math.min(bytesAvailable, maxBufferSize);
    bytesRead = fileInputStream.read(buffer, 0, bufferSize);
}
// multipart 방식으로 파일 전송
dos.writeBytes(lineEnd);
dos.writeBytes(twoHyphens + boundary + twoHyphens + lineEnd);
// 서버로부터 받은 response 값
serverResponseCode = conn.getResponseCode();
String serverResponseMessage = conn.getResponseMessage();
Log.i("uploadFile", "HTTP Response is : "
+ serverResponseMessage + ": " + serverResponseCode);
if (serverResponseCode == 200) {
    runOnUiThread(new Runnable() {
        public void run() {
            Toast.makeText(getApplicationContext(), "File Upload Complete.",
            Toast.LENGTH_SHORT).show();
        }
    });
}
}

```

```

        //close the streams //
        fileInputStream.close();
        dos.flush();
        dos.close();
    } catch (MalformedURLException ex) {
        ex.printStackTrace();
        runOnUiThread(new Runnable() {
            public void run() {
                Toast.makeText(getApplicationContext(), "MalformedURLException",
                    Toast.LENGTH_SHORT).show();
            }
        });
        Log.e("Upload file to server", "error: " + ex.getMessage(), ex);
    } catch (Exception e) {
        e.printStackTrace();
    }
    return serverResponseCode;
} // End else block
}

/**
 * 사용자가 입력한 값으로 수정하기
 *
 * @param fileName
 */
private void setUserDetails(String fileName) {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);

    String userUniqueKey = SaveDataMemberInfo.getAppPreferences(getApplicationContext(),
        "user_key");

    Map<String, String> data = new HashMap<>();
    //이미지변경을 안했을 때
    if (fileName == null) {
        Toast.makeText(getApplicationContext(), fileName, Toast.LENGTH_SHORT).show();
        data.put("user_unique_key", userUniqueKey);
        data.put("name", edtUserName.getText().toString());
        data.put("address", edtAddress.getText().toString());
        data.put("bank_name", edtBankName.getText().toString());
        data.put("account_number", edtBankNumber.getText().toString());
        data.put("account_name", edtAccountName.getText().toString());
        data.put("phone_number", edtPhoneNumber.getText().toString());
        Intent intent = getIntent();
        //판매자 등록할 때 은행명, 계좌번호, 계좌명 RegistSellerActivity로 보내기
        intent.putExtra("bank_name", edtBankName.getText().toString());
    }
}

```

```

        intent.putExtra("account_number", edtBankNumber.getText().toString());
        intent.putExtra("account_name", edtAccountName.getText().toString());
        setResult(RESULT_OK, intent);
        //이미지 변경을 했을 때
    } else {
        data.put("user_unique_key", userUniqueKey);
        data.put("name", edtUserName.getText().toString());
        data.put("address", edtAddress.getText().toString());
        data.put("bank_name", edtBankName.getText().toString());
        data.put("account_number", edtBankNumber.getText().toString());
        data.put("phone_number", edtPhoneNumber.getText().toString());
        data.put("img", imageStorageUrl + fileName);
        data.put("account_name", edtAccountName.getText().toString());
    }
    final Call<UserModifyBean> call = connectService.setUserDetail(data);
    call.enqueue(new Callback<UserModifyBean>() {
        @Override
        public void onResponse(Call<UserModifyBean> call, Response<UserModifyBean>
response) {
            UserModifyBean decode = response.body();
            Log.e("ModifyErr", decode.getErr() + "");
            if (decode.getErr().equals("0")) {
                Toast.makeText(getApplicationContext(), "수정되었습니다.",
Toast.LENGTH_SHORT).show();
            }
        }
        @Override
        public void onFailure(Call<UserModifyBean> call, Throwable t) {
            Log.e("failure", t.getMessage());
        }
    });
}
}

```

PurchaseListActivity.java

```

package sungkyul.ac.kr.leeform.activity.navigation;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import sungkyul.ac.kr.leeform.R;
/**
 * Created by KyungHee on 2016-05-20.
 */

```

```

public class PurchaseListActivity extends AppCompatActivity {
    TextView txtToolBarTitle;
    private Toolbar toolbar;
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_purchase_list);
        toolbar = (Toolbar) findViewById(R.id.toolbarBack);
        toolbar.setContentInsetsAbsolute(0, 0);
        txtToolBarTitle = (TextView) findViewById(R.id.txtToolBarTitle);
        txtToolBarTitle.setText("구매 내역");
        //툴바 완료버튼 보이지 않게 하기
        ImageView tvOK = (ImageView) findViewById(R.id.imgOk);
        tvOK.setVisibility(View.INVISIBLE);
        //뒤로가기 버튼
        ImageView imgBack = (ImageView) findViewById(R.id.imgBackOk);
        imgBack.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
        initializeLayout();
        setListener();
    }
    /**
     * 화면에 보여줄 정보 초기화
     */
    private void initializeLayout() {
    }
    /**
     * 컴포넌트에 Listener 설정
     */
    private void setListener() {
    }
}

```

RegistSellerActivity.java

```

package sungkyul.ac.kr.leeform.activity.navigation;
import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.view.View;
import android.widget.Button;

```

```

import android.widget.ImageView;
import android.widget.Toast;
import java.util.HashMap;
import java.util.Map;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.RegistBean;
import sungkyul.ac.kr.leeform.dto.UserBean;
import sungkyul.ac.kr.leeform.utils.SaveData;
import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo;
import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * Created by YongHoon on 2016-06-01.
 */
public class RegistSellerActivity extends AppCompatActivity {
    Toolbar toolbar;
    Button btnRegistSeller;
    ImageView imgOk, imgBack;
    String URL = StaticURL.BASE_URL;
    String accountNumber;
    String accountName;
    String bankName;
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_regist_seller);
        toolbar = (Toolbar) findViewById(R.id.toolbarBack);
        toolbar.setContentInsetsAbsolute(0, 0);
        imgBack = (ImageView) findViewById(R.id.imgBackOk);
        imgOk = (ImageView) findViewById(R.id.imgOk);
        imgBack.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
        imgOk.setVisibility(View.INVISIBLE);
        btnRegistSeller = (Button) findViewById(R.id.btnRegistSeller);
        btnRegistSeller.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

```



```

        getUserDetails();
    }
});
}
/**
 * 사용자가 은행명,계좌번호,계좌명을 적지 않았을 경우
 * MyPageModifyActivity.class 로 가서 작성하게 함
 * <p>
 * parameter
 * user_key
 */
private void getUserDetails() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    String key = SaveDataMemberInfo.getAppPreferences(getApplicationContext(), "user_key");
    final Call<UserBean> call = connectService.getUserDetail(key);
    call.enqueue(new Callback<UserBean>() {
        @Override
        public void onResponse(Call<UserBean> call, Response<UserBean> response) {
            UserBean decodedResponse = response.body();
            accountNumber =
decodedResponse.getMyinfo_detail().get(0).getAccount_number();
            bankName = decodedResponse.getMyinfo_detail().get(0).getBank_name();
            accountName = decodedResponse.getMyinfo_detail().get(0).getAccount_name();
            if(accountName == null | bankName == null | accountNumber == null){
                Intent intent = new Intent(getApplicationContext(), MyPageModifyActivity.class);
                startActivityForResult(intent, 3000);
                return;
            }
            if (accountName.equals("") | bankName.equals("") | accountNumber.equals("")) {
                Intent intent = new Intent(getApplicationContext(), MyPageModifyActivity.class);
                startActivityForResult(intent, 3000);
                return;
            }
            setUserDetails();
        }
        @Override
        public void onFailure(Call<UserBean> call, Throwable t) {
            Log.e("failure", t.getMessage());
        }
    });
}
/**

```

```

* 사용자가 작성한 은행명,계좌번호,계좌명을 가져온다.
* 가져온 값이 널이 아닌 경우일 때만 setUserDetails() 호출
*
* @param requestCode
* @param resultCode
* @param data
*/
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (resultCode == RESULT_OK) {
        if (requestCode == 3000) {
            bankName = data.getExtras().getString("bank_name");
            accountNumber = data.getExtras().getString("account_number");
            accountName = data.getExtras().getString("account_name");
            if (accountName.equals("") | bankName.equals("") | accountNumber.equals("")) {
                Toast.makeText(getApplicationContext(), "널값저장",
                Toast.LENGTH_SHORT).show();
            } else {
                setUserDetails();
            }
        }
    }
}

/**
* 판매자등록을 한다.
* <p>
* parameter
* key
* accountName
* accountNumber
* bankName
*/
private void setUserDetails() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    String key = SaveDataMemberInfo.getAppPreferences(getApplicationContext(), "user_key");
    Map<String, String> data = new HashMap<>();
    data.put("user_unique_key", key);
    data.put("account_name", accountName);
    data.put("account_number", accountNumber);
    data.put("bank_name", bankName);
    final Call<RegistBean> call = connectService.setResigster(data);

```

```

        call.enqueue(new Callback<RegistBean>() {
            @Override
            public void onResponse(Call<RegistBean> call, Response<RegistBean> response) {
                RegistBean decodedResponse = response.body();
                SaveData.setAppPreferences(getApplicationContext(), "isAuthority","1");
                Toast.makeText(RegistSellerActivity.this, "등록되었습니다.",
                    Toast.LENGTH_SHORT).show();
                finish();
            }
            @Override
            public void onFailure(Call<RegistBean> call, Throwable t) {
                Log.e("failure", t.getMessage());
            }
        });
    }
}

```

SaleListActivity.java

```

package sungkyul.ac.kr.leeform.activity.navigation;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import sungkyul.ac.kr.leeform.R;
/**
 * Created by YongHoon on 2016-06-13.
 */
public class SaleListActivity extends AppCompatActivity {
    TextView txtToolBarTitle;
    private Toolbar toolbar;
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sale_list);
        toolbar = (Toolbar) findViewById(R.id.toolbarBack);
        toolbar.setContentInsetsAbsolute(0, 0);
        txtToolBarTitle = (TextView) findViewById(R.id.txtToolBarTitle);
        txtToolBarTitle.setText("판매 내역");
        //툴바 완료버튼 보이지 않게 하기
        ImageView tvOK = (ImageView) findViewById(R.id.imgOk);
        tvOK.setVisibility(View.INVISIBLE);
        //뒤로가기 버튼
        ImageView imgBack = (ImageView) findViewById(R.id.imgBackOk);
        imgBack.setOnClickListener(new View.OnClickListener() {

```

```

@Override
    public void onClick(View v) {
        finish();
    }
});
initializeLayout();
setListener();
}
/**
 * 화면에 보여줄 정보 초기화
 */
private void initializeLayout() {
}
/**
 * 컴포넌트에 Listener 설정
 */
private void setListener() {
}
}

```

상위패키지	activity	하위패키지	search
KnowHowSearchActivity.java			
<pre> package sungkyul.ac.kr.leeform.activity.search; import android.content.Context; import android.os.Bundle; import android.support.v7.app.AppCompatActivity; import android.support.v7.widget.Toolbar; import android.view.KeyEvent; import android.view.View; import android.view.inputmethod.EditorInfo; import android.view.inputmethod.InputMethodManager; import android.widget.EditText; import android.widget.ImageView; import android.widget.ListView; import android.widget.TextView; import java.util.ArrayList; import retrofit2.Call; import retrofit2.Callback; import retrofit2.Response; import retrofit2.Retrofit; import retrofit2.converter.gson.GsonConverterFactory; import sungkyul.ac.kr.leeform.R; import sungkyul.ac.kr.leeform.adapter.MainListAdapter; import sungkyul.ac.kr.leeform.dao.ConnectService; import sungkyul.ac.kr.leeform.dto.KnowHowBean; import sungkyul.ac.kr.leeform.items.MainListItem; </pre>			

```

import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * Created by YongHoon on 2016-05-23.
 * 노하우 검색
 */
public class KnowHowSearchActivity extends AppCompatActivity implements View.OnClickListener
{
    private Toolbar toolbar;
    private EditText edtSearch;
    private ImageView imgBack, imgSearch;
    private MainListAdapter mainListAdapter;
    private ListView lst;
    private InputMethodManager keyboard;
    ArrayList<MainListItem> listItems = new ArrayList<>();
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_know_how_search);
        layoutSetting();
        toolbar.setContentInsetsAbsolute(0, 0);
        //뒤로가기 버튼
        imgBack.setOnClickListener(this);
        imgSearch.setOnClickListener(this);
        edtSearch.setHint("노하우를 입력하세요.");
        mainListAdapter = new MainListAdapter(getApplicationContext(), R.layout.item_list_main,
listItems);
        lst = (ListView) findViewById(R.id.listKnowHowSearch);
        lst.setAdapter(mainListAdapter);
        edtSearch.setOnEditorActionListener(new TextView.OnEditorActionListener() {
            @Override
            public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
                if (actionId == EditorInfo.IME_ACTION_SEARCH) {
                    listItems.clear();
                    getSearchResult(v.getText().toString());
                    keyboard.hideSoftInputFromWindow(edtSearch.getWindowToken(), 0);
                }
                return false;
            }
        });
    }
    /**
     * 레이아웃 셋팅
     */
    void layoutSetting() {
        imgSearch = (ImageView) findViewById(R.id.imgSearchOk);
        edtSearch = (EditText) findViewById(R.id.edtToolBarSearch);
    }
}

```

```

        imgBack = (ImageView) findViewById(R.id.imgBackSearch);
        toolbar = (Toolbar) findViewById(R.id.toolbarSearch);
        keyboard = (InputMethodManager) getSystemService(Context.INPUT_METHOD_SERVICE);
    }
    @Override
    public void finish() {
        super.finish();
        overridePendingTransition(R.anim.common_slide_from_left,
        R.anim.common_slide_to_right);
    }
    @Override
    public void onClick(View v) {
        switch (v.getId()) {
            case R.id.imgBackSearch: {
                finish();
                break;
            }
            case R.id.imgSearchOk: {
                // 검색
                listItems.clear();
                getSearchResult edtSearch.getText().toString());
                keyboard.hideSoftInputFromWindow(edtSearch.getWindowToken(), 0);
                break;
            }
        }
    }
}

/**
 * 검색한 노하우의 가격, 제작시간, 스크랩 수, 이미지 url, 제목, 간단한 설명 가져오기
 *
 * @param text
 */
private void getSearchResult(String text) {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();
    ConnectService connectService = retrofit.create(ConnectService.class);
    Call<KnowHowBean> call = connectService.getKnowHowSearch(text);
    call.enqueue(new Callback<KnowHowBean>() {
        @Override
        public void onResponse(Call<KnowHowBean> call, Response<KnowHowBean> response)
    {
        KnowHowBean decode = response.body();
        int size;
        if(decode.getErr()=="0") {
            size = decode.getSearch_data().size();

```

```

        } else {
            size = 0;
        }
        for (int i = 0; i < size; i++) {
            listItems.add(new MainListItem(
                Integer.parseInt(decode.getSearch_data().get(i).getWriting_unique_key()),
                decode.getSearch_data().get(i).getPrice(),
                decode.getSearch_data().get(i).getMaking_time(),
                decode.getSearch_data().get(i).getScrap_amount(),
                decode.getSearch_data().get(i).getPicture_url(),
                decode.getSearch_data().get(i).getWriting_name(),
                decode.getSearch_data().get(i).getExplanation()
            ));
        }
        mainListAdapter.notifyDataSetChanged();
    }
    @Override
    public void onFailure(Call<KnowHowBean> call, Throwable t) {
    }
    });
}
}
}

```

MaterialSearchActivity.java

```

package sungkyul.ac.kr.leeform.activity.search;
import android.content.Context;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.KeyEvent;
import android.view.View;
import android.view.inputmethod.EditorInfo;
import android.view.inputmethod.InputMethodManager;
import android.widget.EditText;
import android.widget.GridView;
import android.widget.ImageView;
import android.widget.TextView;
import java.util.ArrayList;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.adapter.MaterialGridAdapter;

```

```

import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.MaterialListBean;
import sungkyul.ac.kr.leeform.items.MaterialGridItem;
import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * Created by KyungHee on 2016-05-22.
 * 재료 검색
 */
public class MaterialSearchActivity extends AppCompatActivity implements View.OnClickListener {
    private EditText edtSearch;
    private GridView grvMaterial;
    private MaterialGridAdapter mAdapter;
    private Toolbar toolbar;
    private ImageView imgBack, imgSearch;
    private InputMethodManager keyboard;
    ArrayList<MaterialGridItem> gridItems = new ArrayList<>();
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_material_search);
        layoutSetting();
        toolbar.setContentInsetsAbsolute(0, 0);
        edtSearch.setHint("재료를 입력하세요.");
        imgSearch.setOnClickListener(this);
        imgBack.setOnClickListener(this);
        grvMaterial = (GridView) findViewById(R.id.grvMaterialSearch);
        mAdapter = new MaterialGridAdapter(getApplicationContext(), R.layout.item_grid_material,
        gridItems);
        grvMaterial.setAdapter(mAdapter);
        edtSearch.setOnEditorActionListener(new TextView.OnEditorActionListener() {
            @Override
            public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
                if (actionId == EditorInfo.IME_ACTION_SEARCH) {
                    gridItems.clear();
                    getSearchResult(v.getText().toString());
                    keyboard.hideSoftInputFromWindow(edtSearch.getWindowToken(), 0);
                    return true;
                }
                return false;
            }
        });
    }
    /**
     * 레이아웃 셋팅
     */
    void layoutSetting() {

```



```

        imgBack = (ImageView) findViewById(R.id.imgBackSearch);
        imgSearch = (ImageView) findViewById(R.id.imgSearchOk);
        edtSearch = (EditText) findViewById(R.id.edtToolBarSearch);
        toolbar = (Toolbar) findViewById(R.id.toolbarSearch);
        keyboard = (InputMethodManager) getSystemService(Context.INPUT_METHOD_SERVICE);
    }

    @Override
    public void finish() {
        super.finish();
        overridePendingTransition(R.anim.common_slide_from_left,
            R.anim.common_slide_to_right);
    }

    @Override
    public void onClick(View v) {
        switch (v.getId()) {
            case R.id.imgBackSearch: {
                finish();
                break;
            }
            case R.id.imgSearchOk: {
                // 검색
                gridItems.clear();
                getSearchResult(edtSearch.getText().toString());
                keyboard.hideSoftInputFromWindow(edtSearch.getWindowToken(), 0);
                break;
            }
        }
    }

    /**
     * 검색한 재료 정보 가져오기
     *
     * @param text
     */
    private void getSearchResult(String text) {
        Retrofit retrofit = new Retrofit.Builder()
            .baseUrl(StaticURL.BASE_URL)
            .addConverterFactory(GsonConverterFactory.create())
            .build();

        ConnectService connectService = retrofit.create(ConnectService.class);
        Call<MaterialListBean> call = connectService.getMaterialSearch(text);
        call.enqueue(new Callback<MaterialListBean>() {
            @Override
            public void onResponse(Call<MaterialListBean> call, Response<MaterialListBean>
                response) {
                MaterialListBean decode = response.body();
                int size;
            }
        });
    }

```

```

        if(decode.getErr()=="0") {
            size = decode.getSearch_data().size();
        } else {
            size = 0;
        }
        for (int i = 0; i < size; i++) {
            gridItems.add(new
MaterialGridItem(decode.getSearch_data().get(i).getMaterial_unique_key(),
decode.getSearch_data().get(i).getMaterial_picture_url(),
decode.getSearch_data().get(i).getMaterial_name(),
decode.getSearch_data().get(i).getMaterial_price()));
        }
        mAdapter.notifyDataSetChanged();
    }
    @Override
    public void onFailure(Call<MaterialListBean> call, Throwable t) {
    }
}
}
}

```

상위패키지	activity	하위패키지	settings
AboutUsActivity.java			
<pre> package sungkyul.ac.kr.leeform.activity.settings; import android.support.v7.app.AppCompatActivity; import android.os.Bundle; import android.support.v7.widget.Toolbar; import android.view.View; import android.widget.ImageView; import android.widget.TextView; import sungkyul.ac.kr.leeform.R; import sungkyul.ac.kr.leeform.adapter.NoticeListAdapter; public class AboutUsActivity extends AppCompatActivity { private ImageView imgBack, imgOk; private TextView txtToolBarTitle; private Toolbar toolbar; @Override protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState); setContentView(R.layout.activity_about_us); settingLayout(); txtToolBarTitle.setText("About Us"); imgBack.setOnClickListener(new View.OnClickListener() { @Override public void onClick(View v) { finish(); } } </pre>			

```

    });
}
private void settingLayout() {
    txtToolBarTitle = (TextView)findViewById(R.id.txtToolBarTitle);
    imgBack = (ImageView)findViewById(R.id.imgBackOk);
    toolbar = (Toolbar)findViewById(R.id.toolbarBack);
    imgOk = (ImageView)findViewById(R.id.imgOk);
    toolbar.setContentInsetsAbsolute(0,0);
    imgOk.setVisibility(View.INVISIBLE);
}
}

```

LicenseActivity.java

```

package sungkyul.ac.kr.leeform.activity.settings;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import java.util.ArrayList;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.adapter.LicenseListAdapter;
import sungkyul.ac.kr.leeform.adapter.NoticeListAdapter;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.LicenseBean;
import sungkyul.ac.kr.leeform.dto.NoticeBean;
import sungkyul.ac.kr.leeform.items.LicenseItem;
import sungkyul.ac.kr.leeform.utils.StaticURL;
public class LicenseActivity extends AppCompatActivity {
    private ImageView imgBack, imgOk;
    private TextView txtToolBarTitle;
    private Toolbar toolbar;
    private ListView lstLicense;
    ArrayList<LicenseItem> licenseItemArrayList = new ArrayList<>();
    LicenseListAdapter licenseListAdapter;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
    }
}

```

```

setContentView(R.layout.activity_license);
layoutSetting();
getLicense();
txtToolBarTitle.setText("라이선스");
imgBack.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        finish();
    }
});
lstLicense.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        Intent it = new Intent(getApplicationContext(), LicenseDetailActivity.class);
        it.putExtra("name", licenseltemArrayList.get(position).getLicenseName());
        it.putExtra("contents", licenseltemArrayList.get(position).getLicenseContents());
        startActivity(it);
    }
});
}

private void layoutSetting() {
    txtToolBarTitle = (TextView)findViewById(R.id.txtToolBarTitle);
    imgBack = (ImageView)findViewById(R.id.imgBackOk);
    toolbar = (Toolbar)findViewById(R.id.toolbarBack);
    imgOk = (ImageView)findViewById(R.id.imgOk);
    lstLicense = (ListView)findViewById(R.id.lstLicense);
    licenseListAdapter = new
LicenseListAdapter(getApplicationContext(), R.layout.item_license, licenseltemArrayList);
    lstLicense.setAdapter(licenseListAdapter);
    toolbar.setContentInsetsAbsolute(0, 0);
    imgOk.setVisibility(View.INVISIBLE);
}

private void getLicense() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();
    ConnectService connectService = retrofit.create(ConnectService.class);
    Call<LicenseBean> call = connectService.getLicense();
    call.enqueue(new Callback<LicenseBean>() {
        @Override
        public void onResponse(Call<LicenseBean> call, Response<LicenseBean> response) {
            LicenseBean decode = response.body();
            for(int i=0; i<decode.getLicense_list().size(); i++) {
                licenseltemArrayList.add(new
LicenseItem(decode.getLicense_list().get(i).getLicense_unique_key(), decode.getLicense_list().get(i).

```

```

        getLicense_name(), decode.getLicense_list().get(i).getLicense_writing_contents());
    }
    licenseListAdapter.notifyDataSetChanged();
}
@Override
public void onFailure(Call<LicenseBean> call, Throwable t) {
}
});
}
}

```

LicenseDetailActivity.java

```

package sungkyul.ac.kr.leeform.activity.settings;
import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.TextView;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.adapter.LicenseListAdapter;
public class LicenseDetailActivity extends AppCompatActivity {
    private ImageView imgBack, imgOk;
    private TextView txtToolBarTitle, txtLicenseDetail;
    private Toolbar toolbar;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_license_detail);
        layoutSetting();
        Intent it = getIntent();
        String name = it.getExtras().getString("name");
        String contents = it.getExtras().getString("contents");
        txtToolBarTitle.setText(name);
        txtLicenseDetail.setText(contents);
        imgBack.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
    }
    private void layoutSetting() {
        txtToolBarTitle = (TextView)findViewById(R.id.txtToolBarTitle);
        txtLicenseDetail = (TextView)findViewById(R.id.txtLicenseDetailContents);
    }
}

```

```

        imgBack = (ImageView)findViewById(R.id.imgBackOk);
        toolbar = (Toolbar)findViewById(R.id.toolbarBack);
        imgOk = (ImageView)findViewById(R.id.imgOk);
        toolbar.setContentInsetsAbsolute(0,0);
        imgOk.setVisibility(View.INVISIBLE);
    }
}

```

NoticeActivity.java

```

package sungkyul.ac.kr.leeform.activity.settings;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.view.View;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.TextView;
import java.util.ArrayList;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.adapter.NoticeListAdapter;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.NoticeBean;
import sungkyul.ac.kr.leeform.items.NoticeItem;
import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * Created by MiSeon on 2016-06-02.
 * 공지사항
 */
public class NoticeActivity extends AppCompatActivity {
    private ListView lstNotice;
    private NoticeListAdapter noticeListAdapter;
    private ImageView imgBack, imgOk;
    TextView txtToolBarTitle;
    Toolbar toolbar;
    ArrayList<NoticeItem> noticeItemArrayList = new ArrayList<>();
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_notice);
        layoutSetting();
        getNotice();
        txtToolBarTitle.setText("공지사항");
    }
}

```

```

        imgBack.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                finish();
            }
        });
    }

    private void layoutSetting() {
        txtToolBarTitle = (TextView)findViewById(R.id.txtToolBarTitle);
        imgBack = (ImageView)findViewById(R.id.imgBackOk);
        toolbar = (Toolbar)findViewById(R.id.toolbarBack);
        imgOk = (ImageView)findViewById(R.id.imgOk);
        toolbar.setContentInsetsAbsolute(0,0);
        imgOk.setVisibility(View.INVISIBLE);
        lstNotice = (ListView) findViewById(R.id.lstNotice);
        noticeListAdapter = new NoticeListAdapter(getApplicationContext(), R.layout.item_notice,
        noticeItemArrayList);
        lstNotice.setAdapter(noticeListAdapter);
    }

    private void getNotice() {
        Retrofit retrofit = new Retrofit.Builder()
            .baseUrl(StaticURL.BASE_URL)
            .addConverterFactory(GsonConverterFactory.create())
            .build();

        ConnectService connectService = retrofit.create(ConnectService.class);
        Call<NoticeBean> call = connectService.getNotice();
        call.enqueue(new Callback<NoticeBean>() {
            @Override
            public void onResponse(Call<NoticeBean> call, Response<NoticeBean> response) {
                NoticeBean decode = response.body();
                for (int i = 0; i < decode.getNotice_list().size(); i++) {
                    noticeItemArrayList.add(new
                    NoticeItem(decode.getNotice_list().get(i).getNotice_unique_key(),
                    decode.getNotice_list().get(i).getNotice_writing_contents(),
                    decode.getNotice_list().get(i).getNotice_date(), decode.getNotice_list().get(i).getNotice_number()));
                }
                noticeListAdapter.notifyDataSetChanged();
            }
            @Override
            public void onFailure(Call<NoticeBean> call, Throwable t) {
            }
        });
    }
}

```

SettingActivity.java

```

package sungkyul.ac.kr.leeform.activity.settings;
import android.content.DialogInterface;

```

```

import android.content.Intent;
import android.os.Bundle;
import android.preference.CheckBoxPreference;
import android.preference.Preference;
import android.preference.PreferenceActivity;
import android.support.annotation.Nullable;
import android.support.v7.app.AlertDialog;
import android.support.v7.widget.Toolbar;
import android.util.Log;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;
import com.kakao.auth.Session;
import com.kakao.network.ErrorResult;
import com.kakao.usermgmt.UserManagement;
import com.kakao.usermgmt.callback.UnLinkResponseCallback;
import com.kakao.util.helper.log.Logger;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.activity.member.LoginActivity;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.AlarmCheckBean;
import sungkyul.ac.kr.leeform.dto.getAlarmStateBean;
import sungkyul.ac.kr.leeform.utils.LoadActivityList;
import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo;
import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * 설정
 * Created by MiSeon on 2016-06-02.
 */
public class SettingActivity extends PreferenceActivity {
    private Toolbar toolbar;
    CheckBoxPreference checkPush;
    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.custom_preference_screen_layout);
        addPreferencesFromResource(R.xml.pref_settings);
        toolbar = (Toolbar) findViewById(R.id.toolbarBack);
        toolbar.setContentInsetsAbsolute(0, 0);
        //툴바 완료버튼 보이지 않게 하기

```



```

ImageView imgOk = (ImageView) findViewById(R.id.imgOk);
imgOk.setVisibility(View.INVISIBLE);
TextView txtToolBarTitle = (TextView) findViewById(R.id.txtToolBarTitle);
txtToolBarTitle.setText("설정");
LoadActivityList.actList.add(SettingActivity.this);
//뒤로가기 버튼
ImageView imgBack = (ImageView) findViewById(R.id.imgBackOk);
imgBack.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        finish();
    }
});
checkBox = (CheckBoxPreference) findPreference("useNotice");
checkBox.setChecked(false);
checkBox.setOnPreferenceChangeListener(new Preference.OnPreferenceChangeListener() {
    @Override
    public boolean onPreferenceChange(Preference preference) {
        changePush();
        return true;
    }
});
Preference leaveLeeForm = findPreference("leaveLeeForm");
leaveLeeForm.setOnPreferenceChangeListener(new Preference.OnPreferenceChangeListener() {
    @Override
    public boolean onPreferenceChange(Preference preference) {
        destory();
        return true;
    }
});
Preference notice = findPreference("notice");
notice.setOnPreferenceChangeListener(new Preference.OnPreferenceChangeListener() {
    @Override
    public boolean onPreferenceChange(Preference preference) {
        startActivity(new Intent(getApplicationContext(), NoticeActivity.class));
        return true;
    }
});
Preference aboutUs = findPreference("aboutUs");
aboutUs.setOnPreferenceChangeListener(new Preference.OnPreferenceChangeListener() {
    @Override
    public boolean onPreferenceChange(Preference preference) {
        startActivity(new Intent(getApplicationContext(), AboutUsActivity.class));
        return true;
    }
});

```

```

Preference license = findPreference("license");
license.setOnPreferenceClickListener(new Preference.OnPreferenceClickListener() {
    @Override
    public boolean onPreferenceClick(Preference preference) {
        startActivity(new Intent(getApplicationContext(), LicenseActivity.class));
        return true;
    }
});
}

private void redirectLoginActivity() {
    Intent it = new Intent(getApplicationContext(), LoginActivity.class);
    Log.e("redi Session ", Session.getCurrentSession() + "");
    startActivity(it);
    LoadActivityList loadActivityList = new LoadActivityList();
    loadActivityList.closeActivity();
}

private void destory() {
    final String appendMessage = getString(R.string.com_kakao_confirm_unlink);
    new AlertDialog.Builder(this)
        .setMessage(appendMessage)
        .setPositiveButton(getString(R.string.com_kakao_ok_button),
            new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    UserManagement.requestUnlink(new UnLinkResponseCallback() {
                        @Override
                        public void onFailure(ErrorResult errorResult) {
                            Logger.e(errorResult.toString());
                        }
                        @Override
                        public void onSessionClosed(ErrorResult errorResult) {
                            redirectLoginActivity();
                        }
                        @Override
                        public void onNotSignedUp() {
                        }
                        @Override
                        public void onSuccess(Long userId) {
                            redirectLoginActivity();
                        }
                    });
                    dialog.dismiss();
                }
            })
        .setNegativeButton(getString(R.string.com_kakao_cancel_button),
            new DialogInterface.OnClickListener() {

```

```

        @Override
        public void onClick(DialogInterface dialog, int which) {
            dialog.dismiss();
        }
    }).show();
}

private void chacngePush() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    Call<AlarmCheckBean> call =
connectService.change(SaveDataMemberInfo.getAppPreferences(getApplicationContext(),"user_key"));
    call.enqueue(new Callback<AlarmCheckBean>() {
        @Override
        public void onResponse(Call<AlarmCheckBean> call, Response<AlarmCheckBean>
response) {
            AlarmCheckBean decode = response.body();
            if(decode.getSet_alarm()==1) {
                checkPush.setChecked(true);
            } else {
                checkPush.setChecked(false);
            }
        }
        @Override
        public void onFailure(Call<AlarmCheckBean> call, Throwable t) {
        }
    });
}

private void checkPush() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    Call<getAlarmStateBean> call =
connectService.checkAlarmState(SaveDataMemberInfo.getAppPreferences(getApplicationContext()
,"user_key"));
    call.enqueue(new Callback<getAlarmStateBean>() {
        @Override
        public void onResponse(Call<getAlarmStateBean> call, Response<getAlarmStateBean>
response) {
            getAlarmStateBean decode = response.body();
            if(decode.getSet_alarm().get(0).getSet_alarm()==1) {
                checkPush.setChecked(true);
            }
        }
    });
}

```

```

        } else {
            checkPush.setChecked(false);
        }
    }
    @Override
    public void onFailure(Call<getAlarmStateBean> call, Throwable t) {
    }
    });
}
}

```

상위패키지	adapter	하위패키지	
CommunityListAdapter.java			
<pre> package sungkyul.ac.kr.leeform.adapter; import android.content.Context; import android.view.LayoutInflater; import android.view.View; import android.view.ViewGroup; import android.widget.BaseAdapter; import android.widget.ImageView; import android.widget.TextView; import com.squareup.picasso.Picasso; import java.util.ArrayList; import sungkyul.ac.kr.leeform.R; import sungkyul.ac.kr.leeform.items.CommunityItem; import sungkyul.ac.kr.leeform.utils.EndString; /** * Created by MiSeon on 2016-05-16. * 커뮤니티 리스트 어댑터 */ public class CommunityListAdapter extends BaseAdapter { private LayoutInflater inflater; private ArrayList<CommunityItem> item; private int layout; public CommunityListAdapter(Context context, int layout, ArrayList<CommunityItem> item) { this.inflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE); this.item = item; this.layout = layout; } /** * Getter */ @Override public int getCount() { return item.size(); } } </pre>			

```

@Override
public Object getItem(int position) {
    return null;
}

@Override
public long getItemId(int position) {
    return position;
}

@Override
public View getView(int position, View convertView, ViewGroup parent) {
    ViewHolder viewHolder;
    if (convertView == null) {
        viewHolder = new ViewHolder();
        convertView = inflater.inflate(layout, parent, false);
        viewHolder.txtUserName = (TextView)
convertView.findViewById(R.id.txtCommunityUserName);
        viewHolder.txtContent = (TextView) convertView.findViewById(R.id.contentCommunity);
        viewHolder.txtReplyCount = (TextView) convertView.findViewById(R.id.txtReplyCount);
        viewHolder.txtImg = (ImageView) convertView.findViewById(R.id.img);
        viewHolder.txtCommunityTime = (TextView)
convertView.findViewById(R.id.txtCommunityTime);
        convertView.setTag(viewHolder);
    } else {
        viewHolder = (ViewHolder) convertView.getTag();
    }
    CommunityItem listItem = item.get(position);
    viewHolder.txtUserName.setText(listItem.getUserName());
    String contents = listItem.getContent();
    if (contents.length() > 70) {
        viewHolder.txtContent.setText(EndString.endString(contents, 70));
    } else {
        viewHolder.txtContent.setText(listItem.getContent());
    }
    viewHolder.txtReplyCount.setText(listItem.getCount());
    viewHolder.txtCommunityTime.setText(listItem.getTime());
    Picasso.with(inflater.getContext()).load(listItem.getImageURL()).into(viewHolder.txtImg);
    return convertView;
}

class ViewHolder {
    private TextView txtUserName;
    private TextView txtContent;
    private TextView txtReplyCount;
    private ImageView txtImg;
    private TextView txtCommunityTime;
}
}

```

CommunityReplyListAdapter.java

```

package sungkyul.ac.kr.leeform.adapter;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.ImageView;
import android.widget.TextView;
import com.squareup.picasso.Picasso;
import java.util.ArrayList;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.items.ReplyItem;
/**
 * Created by misun on 2016-05-18.
 * 커뮤니티 댓글 어댑터
 */
public class CommunityReplyListAdapter extends BaseAdapter {
    private LayoutInflater inflater;
    private ArrayList<ReplyItem> item;
    private int layout;
    public CommunityReplyListAdapter(Context context, int layout, ArrayList<ReplyItem> item) {
        this.inflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        this.item = item;
        this.layout = layout;
    }
    @Override
    public int getCount() {
        return item.size();
    }
    @Override
    public Object getItem(int position) {
        return null;
    }
    @Override
    public long getItemId(int position) {
        return position;
    }
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        ViewHolder viewHolder;
        if (convertView == null) {
            viewHolder = new ViewHolder();
            convertView = inflater.inflate(layout, parent, false);
            viewHolder.userName = (TextView) convertView.findViewById(R.id.rUserName);
            viewHolder.content = (TextView) convertView.findViewById(R.id.rContent);
        }
    }

```

```

        viewHolder.img = (ImageView) convertView.findViewById(R.id.rImg);
        viewHolder.txtReplyTime = (TextView) convertView.findViewById(R.id.txtReplyTime);
        convertView.setTag(viewHolder);
    } else {
        viewHolder = (ViewHolder) convertView.getTag();
    }
    ReplyItem listItem = item.get(position);
    viewHolder.userName.setText(listItem.getrName());
    viewHolder.content.setText(listItem.getrContent());
    Picasso.with(inflater.getContext()).load(listItem.getrImg()).into(viewHolder.img);
    viewHolder.txtReplyTime.setText(listItem.getrTime());
    return convertView;
}

class ViewHolder {
    private TextView userName;
    private TextView content;
    private ImageView img;
    private TextView txtReplyTime;
}
}

```

CreateKnowHowGridAdapter.java

```

package sungkyul.ac.kr.leeform.adapter;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.ImageView;
import com.squareup.picasso.Picasso;
import java.util.ArrayList;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.items.CreateKnowHowItem;
/**
 * Created by HunJin on 2016-05-21.
 * 노하우 작성 상세 어댑터
 */
public class CreateKnowHowGridAdapter extends BaseAdapter {
    private LayoutInflater inflater;
    private ArrayList<CreateKnowHowItem> item;
    private int layout;
    public CreateKnowHowGridAdapter(Context context, int layout, ArrayList<CreateKnowHowItem>
item) {
        this.inflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        this.item = item;
        this.layout = layout;
    }
}

```

```

@Override
public int getCount() {
    return item.size();
}

@Override
public Object getItem(int position) {
    return null;
}

@Override
public long getItemId(int position) {
    return position;
}

@Override
public View getView(int position, View convertView, ViewGroup parent) {
    GridViewHolder gridViewHolder;
    if (convertView == null) {
        gridViewHolder = new GridViewHolder();
        convertView = inflater.inflate(layout, parent, false);
        gridViewHolder.imgGridCreate = (ImageView) convertView.findViewById(R.id.imgCreate);
        convertView.setTag(gridViewHolder);
    } else {
        gridViewHolder = (GridViewHolder) convertView.getTag();
    }
    CreateKnowHowItem gridItem = item.get(position);
    if (gridItem.getmImg() != 0) {
        Picasso.with(inflater.getContext()).load(gridItem.getmImg()).resize(480,
480).centerCrop().into(gridViewHolder.imgGridCreate);
    }
    if (gridItem.getImgUrl() != null) {
        Picasso.with(inflater.getContext()).load("file://" +
gridItem.getImgUrl().trim().toString()).resize(480,
480).centerCrop().error(R.drawable.panza).into(gridViewHolder.imgGridCreate);
    }
    return convertView;
}

class GridViewHolder {
    private ImageView imgGridCreate;
}
}

```

KakaoSDKAdapter.java

```

package sungkyul.ac.kr.leefrom.adapter;
import android.app.Activity;
import android.content.Context;
import com.kakao.auth.ApprovalType;
import com.kakao.auth.AuthType;
import com.kakao.auth.IApplicationConfig;
import com.kakao.auth.ISessionConfig;

```



```

import com.kakao.auth.KakaoAdapter;
import sungkyul.ac.kr.leeform.utils.GlobalApplication;
/**
 * Created by HunJin on 2016-05-26.
 * 카카오 연동을 위한 SDK 어댑터
 */
public class KakaoSDKAdapter extends KakaoAdapter {
    @Override
    public ISessionConfig getSessionConfig() {
        return new ISessionConfig() {
            @Override
            public AuthType[] getAuthTypes() {
                return new AuthType[]{AuthType.KAKAO_TALK};
            }
            @Override
            public boolean isUsingWebviewTimer() {
                return false;
            }
            @Override
            public ApprovalType getApprovalType() {
                return ApprovalType.INDIVIDUAL;
            }
            @Override
            public boolean isSaveFormData() {
                return true;
            }
        };
    }
    @Override
    public IApplicationConfig getApplicationConfig() {
        return new IApplicationConfig() {
            @Override
            public Activity getTopActivity() {
                return GlobalApplication.getCurrentActivity();
            }
            @Override
            public Context getApplicationContext() {
                return GlobalApplication.getGlobalApplicationContext();
            }
        };
    }
}

```

LicenseListAdapter.java

```

package sungkyul.ac.kr.leeform.adapter;
import android.content.Context;
import android.view.LayoutInflater;

```

```

import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.TextView;
import java.util.ArrayList;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.items.LicenseItem;
import sungkyul.ac.kr.leeform.items.NoticeItem;
/**
 * Created by HunJin on 2016-06-13.
 */
public class LicenseListAdapter extends BaseAdapter {
    private LayoutInflater inflater;
    private ArrayList<LicenseItem> item;
    private int layout;
    public LicenseListAdapter(Context context, int layout, ArrayList<LicenseItem> item) {
        this.inflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        this.item = item;
        this.layout = layout;
    }
    @Override
    public int getCount() {
        return item.size();
    }
    @Override
    public Object getItem(int position) {
        return null;
    }
    @Override
    public long getItemId(int position) {
        return position;
    }
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        ViewHolder viewHolder;
        if (convertView == null) {
            viewHolder = new ViewHolder();
            convertView = inflater.inflate(layout, parent, false);
            viewHolder.txtLicenseName = (TextView)
convertView.findViewById(R.id.txtLicenseName);
            convertView.setTag(viewHolder);
        } else {
            viewHolder = (ViewHolder) convertView.getTag();
        }
        LicenseItem listItem = item.get(position);
        viewHolder.txtLicenseName.setText(listItem.getLicenseName().toString());
    }
}

```

```

        return convertView;
    }
    class ViewHolder {
        private TextView txtLicenseName;
    }
}

```

MainFragmentAdapter.java

```

package sungkyul.ac.kr.leeform.adapter;
import android.content.Context;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentManager;
import android.support.v4.app.FragmentPagerAdapter;
import sungkyul.ac.kr.leeform.fragment.CommunityFragment;
import sungkyul.ac.kr.leeform.fragment.HomeFragment;
import sungkyul.ac.kr.leeform.fragment.MaterialFragment;
/**
 * Created by HunJin on 2016-05-10.
 * 가장 기본이 되는 메인 페이지 어댑터
 */
public class MainFragmentAdapter extends FragmentPagerAdapter {
    final int PAGE_COUNT = 3;
    private String tabTitles[] = new String[]{"노하우", "재료", "커뮤니티"};
    private Fragment[] fragments = new Fragment[]{new HomeFragment(), new MaterialFragment(),
    new CommunityFragment()};
    private Context context;
    public MainFragmentAdapter(FragmentManager fm, Context context) {
        super(fm);
        this.context = context;
    }
    @Override
    public Fragment getItem(int position) {
        return fragments[position];
    }
    @Override
    public int getCount() {
        return PAGE_COUNT;
    }
    @Override
    public CharSequence getPageTitle(int position) {
        return tabTitles[position];
    }
}

```

MainListAdapter.java

```

package sungkyul.ac.kr.leeform.adapter;
import android.content.Context;
import android.os.Handler;
import android.text.TextUtils;

```

```

import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.ImageView;
import android.widget.TextView;
import com.squareup.picasso.Picasso;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.OnlyErrBean;
import sungkyul.ac.kr.leeform.items.MainListItem;
import sungkyul.ac.kr.leeform.utils.EndString;
import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo;
import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * Created by HunJin on 2016-05-12.
 * 노하우 리스트 어댑터
 */
public class MainListAdapter extends BaseAdapter {
    private LayoutInflater inflater;
    private ArrayList<MainListItem> item;
    private int layout;
    Handler handler = new Handler();
    String errCode = "0";
    MainListItem listItem;
    public MainListAdapter(Context context, int layout, ArrayList<MainListItem> item) {
        this.inflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        this.item = item;
        this.layout = layout;
    }
    @Override
    public int getCount() {
        return item.size();
    }
    @Override
    public Object getItem(int position) {
        return null;
    }

```

```

    }
    @Override
    public long getItemId(int position) {
        return position;
    }
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        final ViewHolder viewHolder;
        if (convertView == null) {
            viewHolder = new ViewHolder();
            convertView = inflater.inflate(layout, parent, false);
            viewHolder.txtMainCost = (TextView) convertView.findViewById(R.id.txtListCost);
            viewHolder.txtMainTime = (TextView) convertView.findViewById(R.id.txtListTime);
            viewHolder.txtMainLike = (TextView) convertView.findViewById(R.id.txtListLike);
            viewHolder.imgMainList = (ImageView) convertView.findViewById(R.id.imgListItem);
            viewHolder.txtMainName = (TextView) convertView.findViewById(R.id.txtListName);
            viewHolder.txtMainKeyWord = (TextView)
convertView.findViewById(R.id.txtListKeyWord);
            viewHolder.imgMainLike = (ImageView) convertView.findViewById(R.id.imgMainListLike);
            convertView.setTag(viewHolder);
        } else {
            viewHolder = (ViewHolder) convertView.getTag();
        }
        listItem = item.get(position);
        Picasso.with(inflater.getContext()).load(listItem.getUrl()).resize(1020,
492).centerCrop().into(viewHolder.imgMainList);
        checkLike(listItem.getmNumber() + "", viewHolder.imgMainLike);
        viewHolder.txtMainCost.setText(listItem.getmCost());
        viewHolder.txtMainLike.setText(listItem.getmLike());
        viewHolder.txtMainTime.setText(listItem.getmTime());
        viewHolder.txtMainName.setText(EndString.endString(listItem.getmName().toString(),15));

        viewHolder.txtMainKeyWord.setText(EndString.endString(listItem.getmKeyWord().toString(),15));
        viewHolder.imgMainLike.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                checkUncheck(listItem.getmNumber() + "", viewHolder.imgMainLike,
viewHolder.txtMainLike, Integer.parseInt(listItem.getmLike()));
            }
        });
        return convertView;
    }
    class ViewHolder {
        private ImageView imgMainLike;
        private ImageView imgMainList;
        private TextView txtMainLike;
        private TextView txtMainTime;
    }

```

```

private TextView txtMainCost;
private TextView txtMainName;
private TextView txtMainKeyWord;
}

private void checkLike(String writingKey, final ImageView img) {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    Map<String, String> data = new HashMap<>();
    String key = SaveDataMemberInfo.getAppPreferences(inflater.getContext(), "user_key");
    data.put("user_unique_key", key); //user_unique_key 가져오기
    data.put("writing_unique_key", writingKey);
    ConnectService connectService = retrofit.create(ConnectService.class);
    Call<OnlyErrBean> call = connectService.getCheckScrap(data);
    call.enqueue(new Callback<OnlyErrBean>() {
        @Override
        public void onResponse(Call<OnlyErrBean> call, Response<OnlyErrBean> response) {
            OnlyErrBean decodedResponse = response.body();
            errCode = decodedResponse.getErr();
            if (decodedResponse.getErr().equals("3")) {

img.setImageDrawable(inflater.getContext().getResources().getDrawable(R.drawable.main_list_like))
;

                } else {

img.setImageDrawable(inflater.getContext().getResources().getDrawable(R.drawable.main_list_unlik
e));

                }
            }
        @Override
        public void onFailure(Call<OnlyErrBean> call, Throwable t) {
            Log.e("onFailure", t.getMessage());
        }
    });
}

private void likeWrite(final String writingKey, final ImageView img) {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    Map<String, String> data = new HashMap<>();
    String key = SaveDataMemberInfo.getAppPreferences(inflater.getContext(), "user_key");
    data.put("user_unique_key", key); //user_unique_key 가져오기
    data.put("writing_unique_key", writingKey);
    ConnectService connectService = retrofit.create(ConnectService.class);
    Call<OnlyErrBean> call = connectService.setScrap(data);

```

```

        call.enqueue(new Callback<OnlyErrBean>() {
            @Override
            public void onResponse(Call<OnlyErrBean> call, Response<OnlyErrBean> response) {
                OnlyErrBean decodedResponse = response.body();
                errCode = decodedResponse.getErr();

                img.setImageDrawable(inflater.getContext().getResources().getDrawable(R.drawable.main_list_like));
            }

            @Override
            public void onFailure(Call<OnlyErrBean> call, Throwable t) {
                Log.e("onFailure", t.getMessage());
            }
        });
    }

    private void unlikeWrite(String writingKey, final ImageView img) {
        Retrofit retrofit = new Retrofit.Builder()
            .baseUrl(StaticURL.BASE_URL)
            .addConverterFactory(GsonConverterFactory.create())
            .build();

        Map<String, String> data = new HashMap<>();
        String key = SaveDataMemberInfo.getAppPreferences(inflater.getContext(), "user_key");
        data.put("user_unique_key", key); //user_unique_key 가져오기
        data.put("writing_unique_key", writingKey);
        ConnectService connectService = retrofit.create(ConnectService.class);
        Call<OnlyErrBean> call = connectService.unScrap(data);
        call.enqueue(new Callback<OnlyErrBean>() {
            @Override
            public void onResponse(Call<OnlyErrBean> call, Response<OnlyErrBean> response) {
                OnlyErrBean decodedResponse = response.body();
                errCode = decodedResponse.getErr();

                img.setImageDrawable(inflater.getContext().getResources().getDrawable(R.drawable.main_list_unlike));
            }

            @Override
            public void onFailure(Call<OnlyErrBean> call, Throwable t) {
                Log.e("onFailure", t.getMessage());
            }
        });
    }

    private void checkUncheck(final String writingKey, final ImageView img, final TextView txt, final int like) {
        Retrofit retrofit = new Retrofit.Builder()
            .baseUrl(StaticURL.BASE_URL)
            .addConverterFactory(GsonConverterFactory.create())

```

```

        .build();
        Map<String, String> data = new HashMap<>();
        String key = SaveDataMemberInfo.getAppPreferences(inflater.getContext(), "user_key");
        data.put("user_unique_key", key); //user_unique_key 가져오기
        data.put("writing_unique_key", writingKey);
        ConnectService connectService = retrofit.create(ConnectService.class);
        Call<OnlyErrBean> call = connectService.getCheckScrap(data);
        call.enqueue(new Callback<OnlyErrBean>() {
            @Override
            public void onResponse(Call<OnlyErrBean> call, Response<OnlyErrBean> response) {
                OnlyErrBean decodedResponse = response.body();
                errCode = decodedResponse.getErr();
                if (decodedResponse.getErr().equals("3")) {
                    // 삭제 구문
                    unLikeWrite(writingKey, img);
                    listItem.setmLike(like - 1 + "");
                    txt.setText(like - 1 + "");
                } else {
                    // 좋아요 구문
                    likeWrite(writingKey, img);
                    listItem.setmLike(like + 1 + "");
                    txt.setText(like + 1 + "");
                }
            }
            @Override
            public void onFailure(Call<OnlyErrBean> call, Throwable t) {
                Log.e("onFailure", t.getMessage());
            }
        });
    }

    private void push(String writingUniqueKey) {
        Retrofit retrofit = new Retrofit.Builder()
            .baseUrl(StaticURL.BASE_URL)
            .addConverterFactory(GsonConverterFactory.create())
            .build();
        ConnectService connectService = retrofit.create(ConnectService.class);
        Call<OnlyErrBean> call = connectService.push(writingUniqueKey);
        call.enqueue(new Callback<OnlyErrBean>() {
            @Override
            public void onResponse(Call<OnlyErrBean> call, Response<OnlyErrBean> response) {
            }
            @Override
            public void onFailure(Call<OnlyErrBean> call, Throwable t) {
            }
        });
    }
}

```



```

}
MaterialGridAdapter.java
package sungkyul.ac.kr.leeform.adapter;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.ImageView;
import android.widget.TextView;
import com.squareup.picasso.Picasso;
import java.util.ArrayList;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.items.MaterialGridItem;
import sungkyul.ac.kr.leeform.utils.EndString;
/**
 * Created by KyungHee on 2016-05-12.
 * 재료 리스트 어댑터
 */
public class MaterialGridAdapter extends BaseAdapter {
    private LayoutInflater inflater;
    private ArrayList<MaterialGridItem> item;
    private int layout;
    MaterialGridItem gridItem;
    public MaterialGridAdapter(Context context, int layout, ArrayList<MaterialGridItem> item) {
        this.inflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        this.item = item;
        this.layout = layout;
    }
    @Override
    public int getCount() {
        return item.size();
    }
    @Override
    public Object getItem(int position) {
        return null;
    }
    @Override
    public long getItemId(int position) {
        return position;
    }
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        GridViewHolder gridViewHolder;
        if (convertView == null) {
            gridViewHolder = new GridViewHolder();

```

```

        convertView = inflater.inflate(layout, parent, false);
        gridViewHolder.txtGridMaterial = (TextView)
convertView.findViewById(R.id.txtGridMaterialName);
        gridViewHolder.imgGridMaterial = (ImageView)
convertView.findViewById(R.id.imgGridMaterial);
        gridViewHolder.txtGridMaterialCost = (TextView)
convertView.findViewById(R.id.txtGridMaterialCost);
        convertView.setTag(gridViewHolder);
    } else {
        gridViewHolder = (GridViewHolder) convertView.getTag();
    }
    gridItem = item.get(position);

    Picasso.with(inflater.getContext()).load(gridItem.getmUrl()).into(gridViewHolder.imgGridMaterial);
    gridViewHolder.txtGridMaterial.setText(EndString.endString(gridItem.getmName(),12));
    gridViewHolder.txtGridMaterialCost.setText(gridItem.getmPrice());
    return convertView;
}

class GridViewHolder {
    private ImageView imgGridMaterial;
    private TextView txtGridMaterial;
    private TextView txtGridMaterialCost;
}
}

```

MypageFragmentAdapter.java

```

package sungkyul.ac.kr.leeform.adapter;
import android.content.Context;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentManager;
import android.support.v4.app.FragmentPagerAdapter;
import sungkyul.ac.kr.leeform.fragment.MypageScrapFragment;
import sungkyul.ac.kr.leeform.fragment.MypageWriteFragment;
/**
 * Created by user on 2016-06-09.
 */
public class MypageFragmentAdapter extends FragmentPagerAdapter {
    final int PAGE_COUNT = 2;
    private String[] tabTitles = new String[]{"내가 쓴 노하우", "스크랩한 노하우"};
    private Fragment[] fragments = new Fragment[]{new MypageWriteFragment(), new
MypageScrapFragment()};
    private Context context;
    public MypageFragmentAdapter(FragmentManager fm, Context context) {
        super(fm);
        this.context = context;
    }
    @Override
    public Fragment getItem(int position) {

```

```

        return fragments[position];
    }
    @Override
    public int getCount() {
        return PAGE_COUNT;
    }
    @Override
    public CharSequence getPageTitle(int position) {
        return tabTitles[position];
    }
}

```

NoticeListAdapter.java

```

package sungkyul.ac.kr.leeform.adapter;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.TextView;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.items.NoticeItem;
/**
 * Created by HunJin on 2016-06-13.
 */
public class NoticeListAdapter extends BaseAdapter {
    private LayoutInflater inflater;
    private ArrayList<NoticeItem> item;
    private int layout;
    public NoticeListAdapter(Context context, int layout, ArrayList<NoticeItem> item) {
        this.inflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        this.item = item;
        this.layout = layout;
    }
    @Override
    public int getCount() {
        return item.size();
    }
    @Override
    public Object getItem(int position) {
        return null;
    }
    @Override
    public long getItemId(int position) {
        return position;
    }
}

```

```

    }
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        ViewHolder viewHolder;
        if (convertView == null) {
            viewHolder = new ViewHolder();
            convertView = inflater.inflate(layout, parent, false);
            viewHolder.txtNoticeName = (TextView) convertView.findViewById(R.id.txtNoticeName);
            viewHolder.txtNoticeDate = (TextView) convertView.findViewById(R.id.txtNoticeDate);
            convertView.setTag(viewHolder);
        } else {
            viewHolder = (ViewHolder) convertView.getTag();
        }
        NoticeItem listItem = item.get(position);
        viewHolder.txtNoticeName.setText(listItem.getName().toString());
        viewHolder.txtNoticeDate.setText(listItem.getDate().toString());
        return convertView;
    }
    class ViewHolder {
        private TextView txtNoticeName;
        private TextView txtNoticeDate;
    }
}

```

상위패키지	dao	하위패키지	
ConnectService.java			
<pre> package sungkyul.ac.kr.leeform.dao; import java.util.Map; import retrofit2.Call; import retrofit2.Callback; import retrofit2.http.GET; import retrofit2.http.Query; import retrofit2.http.QueryMap; import sungkyul.ac.kr.leeform.dto.AlarmCheckBean; import sungkyul.ac.kr.leeform.dto.CommunityDetailBean; import sungkyul.ac.kr.leeform.dto.CommunityListBean; import sungkyul.ac.kr.leeform.dto.CommunityWritingBean; import sungkyul.ac.kr.leeform.dto.KnowHowBean; import sungkyul.ac.kr.leeform.dto.KnowHowDetailBean; import sungkyul.ac.kr.leeform.dto.KnowHowWritingBean; import sungkyul.ac.kr.leeform.dto.LicenseBean; import sungkyul.ac.kr.leeform.dto.MaterialDetailBean; import sungkyul.ac.kr.leeform.dto.MaterialListBean; import sungkyul.ac.kr.leeform.dto.NoticeBean; import sungkyul.ac.kr.leeform.dto.OnlyErrBean; import sungkyul.ac.kr.leeform.dto.RegistBean; </pre>			

```

import sungkyul.ac.kr.leeform.dto.UserBean;
import sungkyul.ac.kr.leeform.dto.UserInfoBean;
import sungkyul.ac.kr.leeform.dto.UserModifyBean;
import sungkyul.ac.kr.leeform.dto.WritingDetailReplyListBean;
import sungkyul.ac.kr.leeform.dto.getAlarmStateBean;
/**
 * Created by HunJin on 2016-06-10.
 */
public interface ConnectService {
    @GET("community_list.php")
    Call<CommunityListBean> getCommunityList(
        @Query("offset") int offset
    );
    @GET("know_how_writing.php")
    Call<KnowHowWritingBean> setKnowHow(
        @QueryMap Map<String, String> options
    );
    @GET("writing_list.php")
    Call<KnowHowBean> getWritingList(
        @Query("offset") int offset
    );
    @GET("community_writing.php")
    Call<CommunityWritingBean> setCommunity(
        @QueryMap Map<String, String> options
    );
    @GET("community_detail.php")
    Call<CommunityDetailBean> get(
        @Query("community_unique_key") String community_unique_key
    );
    @GET("get.php")
    Call<UserInfoBean> getUserInfo(
        @Query("kakao_unique_key") String kakao_unique_key
    );
    @GET("set.php")
    Call<UserInfoBean> setUserInfo(
        @QueryMap Map<String, String> options
    );
    @GET("know_how_writing_get_key.php")
    Call<KnowHowWritingBean> setKnowGetKey(
        @QueryMap Map<String, String> options
    );
    @GET("reply_community.php")
    Call<CommunityListBean> setCommunityReply(
        @QueryMap Map<String, String> options
    );
    @GET("check_scrap.php")

```

```

Call<OnlyErrBean> getCheckScrap(
    @QueryMap Map<String, String> options
);
@GET("scrap.php")
Call<OnlyErrBean> setScrap(
    @QueryMap Map<String, String> options
);
@GET("cancel_scrap.php")
Call<OnlyErrBean> unScrap(
    @QueryMap Map<String, String> options
);
@GET("writing_detail.php")
Call<KnowHowDetailBean> getKnowHowDetail(
    @Query("writing_unique_key") String writing_unique_key
);
@GET("writing_list_latest.php")
Call<KnowHowBean> getWritingListLatest(
    @Query("offset") int offset
);
@GET("written_by_user_list.php")
Call<KnowHowBean> getMyWriteKnowHow(
    @Query("user_unique_key") String user_unique_key
);
@GET("scrap_list.php")
Call<KnowHowBean> getScrapKnowHow(
    @Query("user_unique_key") String user_unique_key
);
@GET("set_gcm.php")
Call<OnlyErrBean> setGCMTOKEN(
    @QueryMap Map<String, String> options
);
@GET("myinfo_detail.php")
Call<UserBean> getUserDetail(
    @Query("user_unique_key") String user_unique_key
);
@GET("modify_myinfo.php")
Call<UserModifyBean> setUserDetail(
    @QueryMap Map<String, String> options
);
@GET("set_sales_authority.php")
Call<RegistBean> setResigster(
    @QueryMap Map<String, String> options
);
@GET("material_list.php")
Call<MaterialListBean> getMaterialList(
    @Query("offset") int offset

```

```

);
@GET("material_detail.php")
Call<MaterialDetailBean> getMaterialDetail(
    @Query("material_unique_key") String material_unique_key
);
@GET("search.php")
Call<KnowHowBean> getKnowHowSearch(
    @Query("data") String data
);
@GET("search_material.php")
Call<MaterialListBean> getMaterialSearch(
    @Query("data") String data
);
@GET("notice_list.php")
Call<NoticeBean> getNotice();
@GET("license_list.php")
Call<LicenseBean> getLicense();
@GET("PushAlarm.php")
Call<OnlyErrBean> push(
    @Query("writing_unique_key") String writing_unique_key
);
@GET("set_push_alarm.php")
Call<AlarmCheckBean> change(
    @Query("user_unique_key") String user_unique_key
);
@GET("check_push_alarm.php")
Call<getAlarmStateBean> checkAlarmState(
    @Query("user_unique_key") String user_unique_key
);
@GET("view_reply_writing.php")
Call<WritingDetailReplyListBean> getWritingDetailReply(
    @Query("writing_unique_key") String writing_unique_key
);
@GET("reply_writing.php")
Call<OnlyErrBean> setWritingDetailReply(
    @QueryMap Map<String, String> options
);
}

```

상위패키지	dto	하위패키지	
AlarmCheckBean.java			
<pre> package sungkyul.ac.kr.leeform.dto; /** * Created by HunJin on 2016-06-13. */ public class AlarmCheckBean { </pre>			

```

int set_alarm;
int err;
public int getSet_alarm() {
    return set_alarm;
}
public void setSet_alarm(int set_alarm) {
    this.set_alarm = set_alarm;
}
public int getErr() {
    return err;
}
public void setErr(int err) {
    this.err = err;
}
}

```

CommunityDetailBean.java

```

package sungkyul.ac.kr.leeform.dto;
import java.util.List;
import sungkyul.ac.kr.leeform.items.CommunityDetailBeanItem;
import sungkyul.ac.kr.leeform.items.CommunityReplyBeanItem;
/**
 * Created by MiSeon on 2016-05-30.
 * json을 받아줄 클래스를 생성
 * json의 키에 해당하는 값들을 변수 이름으로 지정
 */
public class CommunityDetailBean {
    private String err;
    private String count;
    private List<CommunityDetailBeanItem> community_detail;
    private List<CommunityReplyBeanItem> community_reply;
    public String getErr() {
        return err;
    }
    public void setErr(String err) {
        this.err = err;
    }
    public String getCount() {
        return count;
    }
    public void setCount(String count) {
        this.count = count;
    }
    public List<CommunityDetailBeanItem> getCommunity_detail() {
        return community_detail;
    }
    public void setCommunity_detail(List<CommunityDetailBeanItem> community_detail) {

```



```

        this.community_detail = community_detail;
    }
    public List<CommunityReplyBeanItem> getCommunity_reply() {
        return community_reply;
    }
    public void setCommunity_reply(List<CommunityReplyBeanItem> community_reply) {
        this.community_reply = community_reply;
    }
}

```

CommunityListBean.java

```

package sungkyul.ac.kr.leeform.dto;
import java.util.List;
import sungkyul.ac.kr.leeform.items.CommunityListBeanItem;
/**
 * Created by MiSeon on 2016-05-26.
 * json을 받아줄 클래스를 생성
 * json의 키에 해당하는 값들을 변수 이름으로 지정
 */
public class CommunityListBean {
    private String err;
    private String count;
    private List<CommunityListBeanItem> community_list;
    public String getErr() {
        return err;
    }
    public void setErr(String err) {
        this.err = err;
    }
    public String getCount() {
        return count;
    }
    public void setCount(String count) {
        this.count = count;
    }
    public List<CommunityListBeanItem> getCommunity_list() {
        return community_list;
    }
    public void setCommunity_list(List<CommunityListBeanItem> community_list) {
        this.community_list = community_list;
    }
}

```

CommunityWritingBean.java

```

package sungkyul.ac.kr.leeform.dto;
/**
 * Created by MiSeon on 2016-06-04.
 */

```

```

public class CommunityWritingBean {
    private String err;
    public void setErr(String err) {
        this.err = err;
    }
    public String getErr() {
        return err;
    }
}

```

getAlarmStateBean.java

```

package sungkyul.ac.kr.leeform.dto;
import java.util.List;
import sungkyul.ac.kr.leeform.items.getAlarmStateBeanItem;
/**
 * Created by HunJin on 2016-06-13.
 */
public class getAlarmStateBean {
    int err;
    List<getAlarmStateBeanItem> set_alarm;
    public int getErr() {
        return err;
    }
    public void setErr(int err) {
        this.err = err;
    }
    public List<getAlarmStateBeanItem> getSet_alarm() {
        return set_alarm;
    }
    public void setSet_alarm(List<getAlarmStateBeanItem> set_alarm) {
        this.set_alarm = set_alarm;
    }
}

```

KnowHowBean.java

```

package sungkyul.ac.kr.leeform.dto;
import java.util.List;
import sungkyul.ac.kr.leeform.items.WritingBeanItem;
/**
 * Created by HunJin on 2016-05-25.
 * json을 받아줄 클래스를 생성
 * json의 키에 해당하는 값들을 변수 이름으로 지정
 */
public class KnowHowBean {
    String err;
    String count;
    List<WritingBeanItem> writing_list;
    List<WritingBeanItem> search_data;
}

```

```

    public List<WritingBeanItem> getSearch_data() {
        return search_data;
    }
    public void setSearch_data(List<WritingBeanItem> search_data) {
        this.search_data = search_data;
    }
    public String getErr() {
        return err;
    }
    public void setErr(String err) {
        this.err = err;
    }
    public String getCount() {
        return count;
    }
    public void setCount(String count) {
        this.count = count;
    }
    public List<WritingBeanItem> getWriting_list() {
        return writing_list;
    }
    public void setWriting_list(List<WritingBeanItem> writing_list) {
        this.writing_list = writing_list;
    }
}

```

KnowHowDetailBean.java

```

package sungkyul.ac.kr.leeform.dto;
import java.util.List;
import sungkyul.ac.kr.leeform.items.KnowHowDetailBeanContentsItem;
import sungkyul.ac.kr.leeform.items.KnowHowDetailBeanPictureItem;
/**
 * Created by HunJin on 2016-06-09.
 */
public class KnowHowDetailBean {
    String err;
    List<KnowHowDetailBeanContentsItem> writing_data1;
    List<KnowHowDetailBeanPictureItem> writing_data2;
    public String getErr() {
        return err;
    }
    public void setErr(String err) {
        this.err = err;
    }
    public List<KnowHowDetailBeanContentsItem> getWriting_data1() {
        return writing_data1;
    }
}

```

<pre> public void setWriting_data1(List<KnowHowDetailBeanContentsItem> writing_data1) { this.writing_data1 = writing_data1; } public List<KnowHowDetailBeanPictureItem> getWriting_data2() { return writing_data2; } public void setWriting_data2(List<KnowHowDetailBeanPictureItem> writing_data2) { this.writing_data2 = writing_data2; } } </pre>
KnowHowWritingBean.java
<pre> package sungkyul.ac.kr.leeform.dto; import java.util.List; import sungkyul.ac.kr.leeform.items.KnowHowWritingBeanItem; /** * Created by HunJin on 2016-05-28. * json을 받아줄 클래스를 생성 * json의 키에 해당하는 값들을 변수 이름으로 지정 */ public class KnowHowWritingBean { String err; List<KnowHowWritingBeanItem> writing_unique_key; public List<KnowHowWritingBeanItem> getWriting_unique_key() { return writing_unique_key; } public void setWriting_unique_key(List<KnowHowWritingBeanItem> writing_unique_key) { this.writing_unique_key = writing_unique_key; } public String getErr() { return err; } public void setErr(String err) { this.err = err; } } </pre>
LicenseBean.java
<pre> package sungkyul.ac.kr.leeform.dto; import java.util.List; import sungkyul.ac.kr.leeform.items.LicenseBeanItem; /** * Created by HunJin on 2016-06-13. */ public class LicenseBean { int err; List<LicenseBeanItem> license_list; public int getErr() { </pre>

```

        return err;
    }
    public void setErr(int err) {
        this.err = err;
    }
    public List<LicenseBeanItem> getLicense_list() {
        return license_list;
    }
    public void setLicense_list(List<LicenseBeanItem> license_list) {
        this.license_list = license_list;
    }
}

```

MaterialDetailBean.java

```

package sungkyul.ac.kr.leeform.dto;
import java.util.List;
import sungkyul.ac.kr.leeform.items.MaterialDetailBeanItem;
import sungkyul.ac.kr.leeform.items.MaterialDetailBeanPictureItem;
/**
 * Created by user on 2016-06-12.
 */
public class MaterialDetailBean {
    private String err;
    private String count;
    private List<MaterialDetailBeanItem> material_detail1;
    private List<MaterialDetailBeanPictureItem> material_detail2;
    public String getCount() {
        return count;
    }
    public void setCount(String count) {
        this.count = count;
    }
    public String getErr() {
        return err;
    }
    public void setErr(String err) {
        this.err = err;
    }
    public List<MaterialDetailBeanItem> getMaterial_detail1() {
        return material_detail1;
    }
    public void setMaterial_detail1(List<MaterialDetailBeanItem> material_detail1) {
        this.material_detail1 = material_detail1;
    }
    public List<MaterialDetailBeanPictureItem> getMaterial_detail2() {
        return material_detail2;
    }
}

```

```

        public void setMaterial_detail2(List<MaterialDetailBeanPictureItem> material_detail2) {
            this.material_detail2 = material_detail2;
        }
    }
}

```

MaterialListBean.java

```

package sungkyul.ac.kr.leeform.dto;
import java.util.List;
import sungkyul.ac.kr.leeform.items.MaterialListBeanItem;
/**
 * Created by user on 2016-06-11.
 */
public class MaterialListBean {
    private String err;
    private String count;
    private List<MaterialListBeanItem> material_list;
    private List<MaterialListBeanItem> search_data;
    public List<MaterialListBeanItem> getSearch_data() {
        return search_data;
    }
    public void setSearch_data(List<MaterialListBeanItem> search_data) {
        this.search_data = search_data;
    }
    public String getErr() {
        return err;
    }
    public void setErr(String err) {
        this.err = err;
    }
    public String getCount() {
        return count;
    }
    public void setCount(String count) {
        this.count = count;
    }
    public List<MaterialListBeanItem> getMaterial_list() {
        return material_list;
    }
    public void setMaterial_list(List<MaterialListBeanItem> material_list) {
        this.material_list = material_list;
    }
}

```

NoticeBean.java

```

package sungkyul.ac.kr.leeform.dto;
import java.util.List;
import sungkyul.ac.kr.leeform.items.NoticeBeanItem;
/**

```

```

* Created by HunJin on 2016-06-13.
*/
public class NoticeBean {
    int err;
    List<NoticeBeanItem> notice_list;
    public int getErr() {
        return err;
    }
    public void setErr(int err) {
        this.err = err;
    }
    public List<NoticeBeanItem> getNotice_list() {
        return notice_list;
    }
    public void setNotice_list(List<NoticeBeanItem> notice_list) {
        this.notice_list = notice_list;
    }
}

```

OnlyErrBean.java

```

package sungkyul.ac.kr.leeform.dto;
/**
 * Created by HunJin on 2016-06-07.
 */
public class OnlyErrBean {
    String err;
    public String getErr() {
        return err;
    }
    public void setErr(String err) {
        this.err = err;
    }
}

```

RegistBean.java

```

package sungkyul.ac.kr.leeform.dto;
/**
 * Created by user on 2016-06-11.
 */
public class RegistBean {
    String err;
    public String getErr() {
        return err;
    }
    public void setErr(String err) {
        this.err = err;
    }
}

```

UserBean.java

```
package sungkyul.ac.kr.leeform.dto;
import java.util.List;
import sungkyul.ac.kr.leeform.items.UserBeanItem;
/**
 * Created by user on 2016-06-11.
 */
public class UserBean {
    String err;
    List<UserBeanItem> myinfo_detail;
    public String getErr() {
        return err;
    }
    public void setErr(String err) {
        this.err = err;
    }
    public List<UserBeanItem> getMyinfo_detail() {
        return myinfo_detail;
    }
    public void setMyinfo_detail(List<UserBeanItem> myinfo_detail) {
        this.myinfo_detail = myinfo_detail;
    }
}
```

UserInfoBean.java

```
package sungkyul.ac.kr.leeform.dto;
import java.util.List;
import sungkyul.ac.kr.leeform.items.UserInfoBeanItem;
/**
 * Created by HunJin on 2016-06-01.
 * json을 받아줄 클래스를 생성
 * json의 키에 해당하는 값들을 변수 이름으로 지정
 */
public class UserInfoBean {
    String err;
    List<UserInfoBeanItem> kakao_user_info;
    public List<UserInfoBeanItem> getKakao_user_info() {
        return kakao_user_info;
    }
    public void setKakao_user_info(List<UserInfoBeanItem> kakao_user_info) {
        this.kakao_user_info = kakao_user_info;
    }
    public String getErr() {
        return err;
    }
    public void setErr(String err) {
        this.err = err;
    }
}
```


<pre> } } </pre>
UserModifyBean.java
<pre> package sungkyul.ac.kr.leeform.dto; /** * Created by user on 2016-06-11. */ public class UserModifyBean { String err; String user_unique_key; String name; public String getErr() { return err; } public void setErr(String err) { this.err = err; } public String getUser_unique_key() { return user_unique_key; } public void setUser_unique_key(String user_unique_key) { this.user_unique_key = user_unique_key; } public String getName() { return name; } public void setName(String name) { this.name = name; } } </pre>
WritingDetailReplyListBean.java
<pre> package sungkyul.ac.kr.leeform.dto; import java.util.List; import sungkyul.ac.kr.leeform.items.WritingDetailReplyListItem; /** * Created by HunJin on 2016-06-13. */ public class WritingDetailReplyListBean { int err; List<WritingDetailReplyListItem> reply_list; public int getErr() { return err; } public void setErr(int err) { this.err = err; } } </pre>

```

    public List<WritingDetailReplyListBeanItem> getReply_list() {
        return reply_list;
    }

    public void setReply_list(List<WritingDetailReplyListBeanItem> reply_list) {
        this.reply_list = reply_list;
    }
}

```

상위패키지	fragment	하위패키지
CommunityFragment.java		
<pre> package sungkyul.ac.kr.leeform.fragment; import android.content.Intent; import android.os.Bundle; import android.support.design.widget.FloatingActionButton; import android.support.v4.app.Fragment; import android.support.v4.widget.SwipeRefreshLayout; import android.util.Log; import android.view.LayoutInflater; import android.view.View; import android.view.ViewGroup; import android.widget.AbsListView; import android.widget.AdapterView; import android.widget.AdapterView.OnItemClickListener; import android.widget.ListView; import java.text.ParseException; import java.util.ArrayList; import retrofit2.Call; import retrofit2.Callback; import retrofit2.Response; import retrofit2.Retrofit; import retrofit2.converter.gson.GsonConverterFactory; import sungkyul.ac.kr.leeform.R; import sungkyul.ac.kr.leeform.activity.community.CommunityCreateActivity; import sungkyul.ac.kr.leeform.activity.community.CommunityDetailActivity; import sungkyul.ac.kr.leeform.adapter.CommunityListAdapter; import sungkyul.ac.kr.leeform.dao.ConnectService; import sungkyul.ac.kr.leeform.dto.CommunityListBean; import sungkyul.ac.kr.leeform.items.CommunityItem; import sungkyul.ac.kr.leeform.utils.StaticURL; import sungkyul.ac.kr.leeform.utils.TimeTransForm; /** * Created by MiSeon on 2016-05-10. * 커뮤니티 리스트가 뜬다. */ public class CommunityFragment extends Fragment { private int check = 0; private View cView; </pre>		

```

private CommunityListAdapter adapter;
private static String URL = StaticURL.BASE_URL;
ListView lst;
ArrayList<CommunityItem> listItem;
FloatingActionButton fab1;
Intent intent;
int offset = 0;
int count = 0;
boolean is_scroll = true;
boolean is_refresh = true;
private SwipeRefreshLayout swipeRefreshLayout;
private boolean isScrollingUp = false;
private int mLastFirstVisibleItem = 0;
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)
{
    super.onCreateView(inflater, container, savedInstanceState);
    //inflater를 통해 xml을 가져온다.
    cView = inflater.inflate(R.layout.fragment_community, container, false);
    layoutSetting();
    setListener();
    init();
    getCommunityDetailList();
    return cView;
}
/**
 * Layout Setting
 */
public void layoutSetting() {
    listItem = new ArrayList<>();
    //adapter를 통해 xml을 ArrayList에 설정한다.
    adapter = new CommunityListAdapter(getContext(), R.layout.item_list_community, listItem);
    lst = (ListView) cView.findViewById(R.id.listCommunity);
    fab1 = (FloatingActionButton) cView.findViewById(R.id.fab1);
    swipeRefreshLayout = (SwipeRefreshLayout)
cView.findViewById(R.id.communitySwipeRefresh);
}
private void setListener() {
    //lst에 adapter를 등록한다.
    lst.setAdapter(adapter);
    //커뮤니티 목록 중 선택한 넘버 보내기
    lst.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<> parent, View view, int position, long id) {
            intent = new Intent(getContext(), CommunityDetailActivity.class);
            Log.e("Number", listItem.get(position).getcNumber() + "");
        }
    });
}

```

```

        /*intent.putExtra("Number",listItem.get(position).getcNumber());*/
        intent.putExtra("Number", listItem.get(position).getcNumber());
        intent.putExtra("image", listItem.get(position).getcImageUrl());
        intent.putExtra("name", listItem.get(position).getcName());
        intent.putExtra("time", listItem.get(position).getcTime());
        intent.putExtra("contents", listItem.get(position).getcContent());
        startActivity(intent);
    }
});
lst.setOnScrollListener(new AbsListView.OnScrollListener() {
    @Override
    public void onScrollStateChanged(AbsListView view, int scrollState) {
        final ListView lw = (ListView) view;
        if (view.getId() == lw.getId()) {
            final int currentFirstVisibleItem = lw.getFirstVisiblePosition();
            if (currentFirstVisibleItem > mLastFirstVisibleItem) {
                isScrollingUp = false;
                fab1.hide();
            } else if (currentFirstVisibleItem < mLastFirstVisibleItem) {
                isScrollingUp = true;
                fab1.show();
            }
            mLastFirstVisibleItem = currentFirstVisibleItem;
        }
    }
    @Override
    public void onScroll(AbsListView view, int firstVisibleItem, int visibleItemCount, int
totalItemCount) {
        if (firstVisibleItem + visibleItemCount == totalItemCount) {
            if (count != 0 && offset % 6 == 0) {
                if (is_scroll) {
                    is_scroll = false;
                    is_refresh = false;
                    getCommunityDetailList();
                }
            }
        }
    }
});
//작성버튼 클릭시 커뮤니티작성화면으로 이동
fab1.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        startActivity(new Intent(getActivity(), CommunityCreateActivity.class));
    }
});

```

```

swipeRefreshLayout.setOnRefreshListener(new SwipeRefreshLayout.OnRefreshListener() {
    @Override
    public void onRefresh() {
        init();
        getCommunityDetailList();
    }
});
}
@Override
public void onDestroyView() {
    super.onDestroyView();
    // 다른 프래그먼트 가면 초기화
    check = 1;
    init();
}
/**
 * 커뮤니티의 정보를 가져와
 * ArrayList에 저장
 */
public void getCommunityDetailList() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    Call<CommunityListBean> call = connectService.getCommunityList(offset);
    call.enqueue(new Callback<CommunityListBean>() {
        @Override
        public void onResponse(Call<CommunityListBean> call, Response<CommunityListBean>
response) {
            CommunityListBean decode = response.body(); //CommunityListBean 형식으로
디코딩

            Log.e("err", decode.getErr());
            //커뮤니티 목록 개수만큼 list에 CommunityItem(작성자이름, 댓글개수, 커뮤니티 내용,
작성자이미지) 추가
            for (int i = 0; i < Integer.parseInt(decode.getCount()); i++) {
                String date = decode.getCommunity_list().get(i).getCommunity_writing_date();
                try {
                    date = TimeTransForm.formatTimeString(date);
                } catch (ParseException e) {
                    e.printStackTrace();
                }
                listItem.add(new
CommunityItem(decode.getCommunity_list().get(i).getCommunity_unique_key(),
decode.getCommunity_list().get(i).getName(),
decode.getCommunity_list().get(i).getReply_community_amount(),
decode.getCommunity_list().get(i).getCommunity_writing_contents(),
decode.getCommunity_list().get(i).getImg(), date));

```

```

        }
        offset += Integer.parseInt(decode.getCount());
        count += Integer.parseInt(decode.getCount());
        swipeRefreshLayout.setRefreshing(false);
        is_scroll = true;
        adapter.notifyDataSetChanged();
    }
    @Override
    public void onFailure(Call<CommunityListBean> call, Throwable t) {
        Log.e("failure", t.getMessage());
    }
}
});
}
void init() {
    offset = 0;
    count = 0;
    is_scroll = true;
    is_refresh = true;
    listItem.clear();
}
}

```

HomeFragment.java

```

package sungkyul.ac.kr.leeform.fragment;
import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.design.widget.FloatingActionButton;
import android.support.v4.app.Fragment;
import android.support.v4.widget.SwipeRefreshLayout;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AbsListView;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Spinner;
import android.widget.Toast;
import java.util.ArrayList;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;

```

```

import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.activity.knowhow.DetailKnowHowActivity;
import sungkyul.ac.kr.leeform.activity.knowhow.WriteKnowHowActivity;
import sungkyul.ac.kr.leeform.adapter.MainListAdapter;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.KnowHowBean;
import sungkyul.ac.kr.leeform.items.MainListItem;
import sungkyul.ac.kr.leeform.utils.SaveData;
import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * Created by HunJin on 2016-05-10.
 * 노하우리스트화면
 */
public class HomeFragment extends Fragment {
    private int check = 0;
    private View mView;
    int offset = 0;
    int count = 0;
    boolean is_scroll = true;
    boolean is_refresh = true;
    boolean sort = true; // sort = true : 최신순, sort = false : 인기순
    private MainListAdapter adapter;
    private Spinner mSpinnerCategory, mSpinnerSort;
    private FloatingActionButton fab;
    private boolean isScrollingUp = false;
    private int mLastFirstVisibleItem = 0;
    private SwipeRefreshLayout swipeRefreshLayout;
    ListView lst;
    private static String URL = StaticURL.BASE_URL;
    ArrayList<MainListItem> listItem = new ArrayList<>();
    @Override
    public void onResume() {
        super.onResume();
        if (SaveData.getAppPreferences(getActivity().getApplicationContext(),
"isAuthority").equals("1")) {
            fab.setVisibility(View.VISIBLE);
        }else{
            fab.setVisibility(View.INVISIBLE);
        }
    }
    @Override
    public void onDestroyView() {
        super.onDestroyView();
        // 다른 프래그먼트 가면 초기화
        check = 1;
        init();
    }

```

```

    }

    private void initializeLayout() {
        adapter = new MainListAdapter(getContext(), R.layout.item_list_main, listItem);
        mSpinnerCategory = (Spinner) mView.findViewById(R.id.spnKnowCategory);
        //(Spinner)findViewById(R.id.spnKnowCategory); 오류=>보여줄 View가 없기 때문
        mSpinnerSort = (Spinner) mView.findViewById(R.id.spnKnowSort);
        String[] mCategory = getResources().getStringArray(R.array.category); //카테고리의 내용들을
        배열(mCategory)에 저장
        String[] mSort = getResources().getStringArray(R.array.sort); //정렬의 내용들을 배열(mSort)에
        저장
        ArrayAdapter<String> mSpinnerCategoryAdapter = new ArrayAdapter<String>(getContext(),
        R.layout.item_spinner, mCategory);
        //기본으로 제공하는 layout(simple_spinner_item), 리스트에 있는 내용을 mCategory의 내용으로
        채우기 위해 ArrayAdapter 이용
        ArrayAdapter<String> mSpinnerSortAdapter = new ArrayAdapter<String>(getContext(),
        R.layout.item_spinner, mSort);

        mSpinnerCategoryAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown
        _item);
        //adapter를 통해 리스트 형태로 늘리는 메소드

        mSpinnerSortAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_ite
        m);

        mSpinnerCategory.setAdapter(mSpinnerCategoryAdapter); //스피너에 adapter 설정
        mSpinnerSort.setAdapter(mSpinnerSortAdapter);
        lst = (ListView) mView.findViewById(R.id.listMain);
        lst.setAdapter(adapter);
        fab = (FloatingActionButton) mView.findViewById(R.id.fab); //작성하기 버튼
        if(SaveData.getAppPreferences(getContext(), "isAuthority").equals("1")) {
            fab.setVisibility(View.VISIBLE);
        } else {
            fab.setVisibility(View.INVISIBLE);
        }
    }

    private void initializeList() {
        // 초기화하고 아이템가져오기
        if (sort == true) {
            latestParsing();
        } else {
            leeformParsing();
        }
    }

    private void setListener() {
        fab.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                startActivity(new Intent(getActivity(), WriteKnowHowActivity.class));
            }
        });
    }

```



```

});
swipeRefreshLayout = (SwipeRefreshLayout) mView.findViewById(R.id.swipe_refresh_widget);
swipeRefreshLayout.setOnRefreshListener(new SwipeRefreshLayout.OnRefreshListener() {
    @Override
    public void onRefresh() {
        init();
        if (sort == true) {
            latestParsing();
        } else {
            leeformParsing();
        }
    }
});
lst.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    //리스트의 아이템 선택했을 때
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
        Intent itKnowhowDetail = new Intent(getActivity(), DetailKnowHowActivity.class);
        itKnowhowDetail.putExtra("image", listItem.get(position).getmUrl() + "");
        itKnowhowDetail.putExtra("knowhowkey", listItem.get(position).getmNumber() + "");
        startActivity(itKnowhowDetail);
    }
});
lst.setOnScrollListener(new AbsListView.OnScrollListener() {
    @Override
    public void onScrollStateChanged(AbsListView view, int scrollState) {
        final ListView lw = (ListView) view;
        if (view.getId() == lw.getId()) {
            final int currentFirstVisibleItem = lw.getFirstVisiblePosition();
            if (currentFirstVisibleItem > mLastFirstVisibleItem) {
                isScrollingUp = false;
                fab.hide();
            } else if (currentFirstVisibleItem < mLastFirstVisibleItem &&
SaveData.getAppPreferences(getActivity(), "isAuthority").equals("1")) {
                isScrollingUp = true;
                fab.show();
            }
            mLastFirstVisibleItem = currentFirstVisibleItem;
        }
    }
    @Override
    public void onScroll(AbsListView view, int firstVisibleItem, int visibleItemCount, int
totalItemCount) {
        if (firstVisibleItem + visibleItemCount == totalItemCount) {
            if (count != 0 && offset % 6 == 0) {
                if (is_scroll) {

```

```

        is_scroll = false;
        is_refresh = false;
        if (sort == true) {
            latestParsing();
        } else {
            leeformParsing();
        }
    }
}

});
mSpinnerCategory.setOnTouchListener(new View.OnTouchListener() {
    @Override
    public boolean onTouch(View view, MotionEvent motionEvent) {
        check++;
        return false;
    }
});
mSpinnerSort.setOnTouchListener(new View.OnTouchListener() {
    @Override
    public boolean onTouch(View view, MotionEvent motionEvent) {
        check++;
        return false;
    }
});
mSpinnerCategory.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
//카테고리 아이템 선택했을 때
    @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position, long id)
    {
        // 처음에는 실행 안되게
        if (check > 0) {
            Toast.makeText(getActivity(), parent.getItemAtPosition(position) + "",
            Toast.LENGTH_SHORT).show();
        }
    }
    @Override
    public void onNothingSelected(AdapterView<?> parent) {
    }
});
mSpinnerSort.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() { //정렬
아이템 선택했을 때
    @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position, long id)
    {
        // 처음에는 실행 안되게

```

```

        if (check > 0) {
            if (parent.getItemAtPosition(position).equals("인기순")) {
                sort = false;
                init();
                initializeList();
            } else {
                sort = true;
                init();
                initializeList();
            }
        }
    }

    @Override
    public void onNothingSelected(AdapterView<?> parent) {
    }

});

}

@Nullable
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)
{
    super.onCreateView(inflater, container, savedInstanceState);
    mView = inflater.inflate(R.layout.fragment_home, container, false);
    initializeLayout();
    init();
    initializeList();
    setListener();
    return mView;
}

void init() {
    offset = 0;
    count = 0;
    check = 0;
    is_scroll = true;
    is_refresh = true;
    listItem.clear();
}

private void leeformParsing() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    Call<KnowHowBean> call = connectService.getWritingList(offset);
    call.enqueue(new Callback<KnowHowBean>() {
        @Override

```

```

        public void onResponse(Call<KnowHowBean> call, Response<KnowHowBean> response)
    {
        KnowHowBean decode = response.body();
        for (int i = 0; i < Integer.parseInt(decode.getCount()); i++) {
            listItem.add(new
MainListItem(Integer.parseInt(decode.getWriting_list().get(i).getWriting_unique_key()),
decode.getWriting_list().get(i).getCost(),
                decode.getWriting_list().get(i).getMaking_time(),
                decode.getWriting_list().get(i).getScrap_amount(),
                decode.getWriting_list().get(i).getPicture_url(),
                decode.getWriting_list().get(i).getWriting_name(),
                decode.getWriting_list().get(i).getExplanation()
            ));
        }
        offset += Integer.parseInt(decode.getCount());
        count += Integer.parseInt(decode.getCount());
        swipeRefreshLayout.setRefreshing(false);
        is_scroll = true;
        adapter.notifyDataSetChanged();
    }
    @Override
    public void onFailure(Call<KnowHowBean> call, Throwable t) {
        Log.e("failure", t.getMessage());
    }
    });
}

private void latestParsing() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();
    ConnectService connectService = retrofit.create(ConnectService.class);
    Call<KnowHowBean> call = connectService.getWritingListLatest(offset);
    call.enqueue(new Callback<KnowHowBean>() {
        @Override
        public void onResponse(Call<KnowHowBean> call, Response<KnowHowBean> response)
    {
        KnowHowBean decode = response.body();
        for (int i = 0; i < Integer.parseInt(decode.getCount()); i++) {
            listItem.add(new
MainListItem(Integer.parseInt(decode.getWriting_list().get(i).getWriting_unique_key()),
decode.getWriting_list().get(i).getCost(),
                decode.getWriting_list().get(i).getMaking_time(),
                decode.getWriting_list().get(i).getScrap_amount(),
                decode.getWriting_list().get(i).getPicture_url(),
                decode.getWriting_list().get(i).getWriting_name(),
                decode.getWriting_list().get(i).getExplanation()
            ));
        }
    }
    });
}

```

```

        ));
    }
    offset += Integer.parseInt(decode.getCount());
    count += Integer.parseInt(decode.getCount());
    swipeRefreshLayout.setRefreshing(false);
    is_scroll = true;
    adapter.notifyDataSetChanged();
}
@Override
public void onFailure(Call<KnowHowBean> call, Throwable t) {
    Log.e("failure", t.getMessage());
}
});
}
}

```

MaterialFragment.java

```

package sungkyul.ac.kr.leeform.fragment;
import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.Nullable;
import android.support.v4.app.Fragment;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AbsListView;
import android.widget.AdapterView;
import android.widget.GridView;
import java.util.ArrayList;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.activity.material.MaterialDetailActivity;
import sungkyul.ac.kr.leeform.adapter.MaterialGridAdapter;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.MaterialListBean;
import sungkyul.ac.kr.leeform.items.MaterialGridItem;
import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * Created by KyungHee on 2016-05-12.
 */
public class MaterialFragment extends Fragment {
    private int check = 0;

```

```

private View mView;
int offset = 0;
int count = 0;
boolean is_scroll = true;
private boolean isScrollingUp = false;
private GridView grvMaterial;
private MaterialGridAdapter mAdapter;
GridView gridView;
private int mLastFirstVisibleItem = 0;
Intent intent;
private static String URL = StaticURL.BASE_URL;
ArrayList<MaterialGridItem> gridItems = new ArrayList<>();
@Override
public void onDestroyView() {
    super.onDestroyView();
    // 다른 프래그먼트 가면 초기화
    check = 1;
    init();
}
@Nullable
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)
{
    super.onCreateView(inflater, container, savedInstanceState);
    mView = inflater.inflate(R.layout.fragment_material, container, false);
    layoutSetting();
    setListener();
    init();
    getMaterialDetailList();
    gridView.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
            Intent it = new Intent(getActivity(), MaterialDetailActivity.class);
            Log.e("material_unique_key", gridItems.get(position).getmKey() + "");
            it.putExtra("material_unique_key", gridItems.get(position).getmKey() + "");
            startActivity(it);
        }
    });
    return mView;
}
public void layoutSetting() {
    gridItems = new ArrayList<>();
    //adapter를 통해 xml을 ArrayList에 설정한다.
    mAdapter = new MaterialGridAdapter(getContext(), R.layout.item_grid_material, gridItems);
    gridView = (GridView) mView.findViewById(R.id.grvMaterial);
}

```

```

private void setListener() {
    //lst에 adapter를 등록한다.
    gridView.setAdapter(mAdapter);
    gridView.setOnScrollListener(new AbsListView.OnScrollListener() {
        @Override
        public void onScrollStateChanged(AbsListView view, int scrollState) {
            final GridView gv = (GridView) view;
            if (view.getId() == gv.getId()) {
                final int currentFirstVisibleItem = gv.getFirstVisiblePosition();
                if (currentFirstVisibleItem > mLastFirstVisibleItem) {
                    isScrollingUp = false;
                } else if (currentFirstVisibleItem < mLastFirstVisibleItem) {
                    isScrollingUp = true;
                }
                mLastFirstVisibleItem = currentFirstVisibleItem;
            }
        }
        @Override
        public void onScroll(AbsListView view, int firstVisibleItem, int visibleItemCount, int
totalItemCount) {
            if (firstVisibleItem + visibleItemCount == totalItemCount) {
                if (count != 0 && offset % 6 == 0) {
                    if (is_scroll) {
                        is_scroll = false;
                        getMaterialDetailList();
                    }
                }
            }
        }
    });
}

public void getMaterialDetailList() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    Call<MaterialListBean> call = connectService.getMaterialList(offset);
    call.enqueue(new Callback<MaterialListBean>() {
        @Override
        public void onResponse(Call<MaterialListBean> call, Response<MaterialListBean>
response) {
            MaterialListBean decode = response.body(); //CommunityListBean 형식으로 디코딩
            Log.e("err", decode.getErr());
            //커뮤니티 목록 개수만큼 lst에 CommunityItem(작성자이름, 댓글개수, 커뮤니티 내용,
작성자이미지) 추가

```

```

        for (int i = 0; i < Integer.parseInt(decode.getCount()); i++) {
            gridItems.add(new
MaterialGridItem(decode.getMaterial_list().get(i).getMaterial_unique_key(),
decode.getMaterial_list().get(i).getMaterial_picture_url(),
decode.getMaterial_list().get(i).getMaterial_name(),
decode.getMaterial_list().get(i).getMaterial_price()));
        }
        offset += Integer.parseInt(decode.getCount());
        count += Integer.parseInt(decode.getCount());
        is_scroll = true;
        mAdapter.notifyDataSetChanged();
    }
    @Override
    public void onFailure(Call<MaterialListBean> call, Throwable t) {
        Log.e("failure", t.getMessage());
    }
});
}
void init() {
    offset = 0;
    count = 0;
    check = 0;
    is_scroll = true;
    gridItems.clear();
}
@Override
public void onResume() {
    super.onResume();
    mAdapter.notifyDataSetChanged();
}
}
}

```

MypageScrapFragment.java

```

package sungkyul.ac.kr.leefrom.fragment;
import android.content.Intent;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.AdapterView;
import android.widget.ListView;
import java.util.ArrayList;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;

```



```

import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.activity.knowhow.DetailKnowHowActivity;
import sungkyul.ac.kr.leeform.adapter.MainListAdapter;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.KnowHowBean;
import sungkyul.ac.kr.leeform.items.MainListItem;
import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo;
import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * Created by user on 2016-06-10.
 */
public class MypageScrapFragment extends Fragment {
    private int check = 0;
    private View mView;
    private MainListAdapter adapter;
    private static String URL = StaticURL.BASE_URL;
    ArrayList<MainListItem> listItem = new ArrayList<>();
    @Override
    public void onDestroyView() {
        super.onDestroyView();
        // 다른 프래그먼트 가면 초기화
        check = 1;
    }
    @Override
    public void onResume() {
        super.onResume();
        // 다른 프래그먼트에 갔다가 오면
        if (check == 1) {
            check = 0;
        } else {
            check = 1;
        }
        leeformParsing();
    }
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
        super.onCreateView(inflater, container, savedInstanceState);
        mView = inflater.inflate(R.layout.fragment_mypage, container, false);
        adapter = new MainListAdapter(getContext(), R.layout.item_list_main, listItem);
        ListView lst = (ListView) mView.findViewById(R.id.listMain);
        lst.setAdapter(adapter);
        lst.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            //리스트의 아이템 선택했을 때
            @Override

```

```

        public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
            Intent itKnowhowDetail = new Intent(getActivity(), DetailKnowHowActivity.class);
            Log.e("img", listItem.get(position).getmUrl());
            itKnowhowDetail.putExtra("image", listItem.get(position).getmUrl() + "");
            itKnowhowDetail.putExtra("knowhowkey", listItem.get(position).getmNumber() + "");
            startActivity(itKnowhowDetail);
        }
    });
    leeformParsing();
    return mView;
}

//스크랩한 노하우 가져오기
private void leeformParsing() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    String user_key = SaveDataMemberInfo.getAppPreferences(getContext(), "user_key");
    Call<KnowHowBean> call = connectService.getScrapKnowHow(user_key);
    call.enqueue(new Callback<KnowHowBean>() {
        @Override
        public void onResponse(Call<KnowHowBean> call, Response<KnowHowBean> response)
        {
            KnowHowBean decode = response.body();
            Log.e("err", decode.getErr());
            listItem.clear();
            for (int i = 0; i < Integer.parseInt(decode.getCount()); i++) {
                listItem.add(new
MainListItem(Integer.parseInt(decode.getWriting_list().get(i).getWriting_unique_key()),
decode.getWriting_list().get(i).getPrice(),
                decode.getWriting_list().get(i).getMaking_time(),
                decode.getWriting_list().get(i).getScrap_amount(),
                decode.getWriting_list().get(i).getPicture_url(),
                decode.getWriting_list().get(i).getWriting_name(),
                decode.getWriting_list().get(i).getExplanation()

            ));
        }
        adapter.notifyDataSetChanged();
    }
    @Override
    public void onFailure(Call<KnowHowBean> call, Throwable t) {
        Log.e("failure", t.getMessage());
    }
});
}

```

```

}
MypageWriteFragment.java
package sungkyul.ac.kr.leeform.fragment;
import android.content.Intent;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.AdapterView;
import android.widget.ListView;
import java.util.ArrayList;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;
import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.activity.knowhow.DetailKnowHowActivity;
import sungkyul.ac.kr.leeform.adapter.MainListAdapter;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.KnowHowBean;
import sungkyul.ac.kr.leeform.items.MainListItem;
import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo;
import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * Created by user on 2016-06-10.
 */
public class MypageWriteFragment extends Fragment {
    private int check = 0;
    private View mView;
    private MainListAdapter adapter;
    private static String URL = StaticURL.BASE_URL;
    ArrayList<MainListItem> listItem = new ArrayList<>();
    @Override
    public void onDestroyView() {
        super.onDestroyView();
        // 다른 프래그먼트 가면 초기화
        check = 1;
    }
    @Override
    public void onResume() {
        super.onResume();
        // 다른 프래그먼트에 갔다가 오면
        if (check == 1) {

```

```

        check = 0;
    } else {
        check = 1;
    }
    leeformParsing();
}

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)
{
    super.onCreateView(inflater, container, savedInstanceState);
    mView = inflater.inflate(R.layout.fragment_mypage, container, false);
    adapter = new MainListAdapter(getContext(), R.layout.item_list_main, listItem);
    ListView lst = (ListView) mView.findViewById(R.id.listMain);
    lst.setAdapter(adapter);
    lst.setOnItemClickListener(new AdapterView.OnItemClickListener() {
        //리스트의 아이템 선택했을 때
        @Override
        public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
            Intent itKnowhowDetail = new Intent(getActivity(), DetailKnowHowActivity.class);
            Log.e("img", listItem.get(position).getmUrl());
            itKnowhowDetail.putExtra("image", listItem.get(position).getmUrl() + "");
            itKnowhowDetail.putExtra("knowhowkey", listItem.get(position).getmNumber() + "");
            startActivity(itKnowhowDetail);
        }
    });
    leeformParsing();
    return mView;
}

//내가 쓴 노하우 가져오기
private void leeformParsing() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    ConnectService connectService = retrofit.create(ConnectService.class);
    String user_key = SaveDataMemberInfo.getAppPreferences(getContext(), "user_key");
    Call<KnowHowBean> call = connectService.getMyWriteKnowHow(user_key);
    call.enqueue(new Callback<KnowHowBean>() {
        @Override
        public void onResponse(Call<KnowHowBean> call, Response<KnowHowBean> response)
    {
        KnowHowBean decode = response.body();
        Log.e("err", decode.getErr());
        listItem.clear();
        for (int i = 0; i < Integer.parseInt(decode.getCount()); i++) {
            Log.e("imgUrl", decode.getWriting_list().get(i).getPicture_url());

```

```

        listItem.add(new
MainListItem(Integer.parseInt(decode.getWriting_list().get(i).getWriting_unique_key()),
decode.getWriting_list().get(i).getPrice(),
                decode.getWriting_list().get(i).getMaking_time(),
                decode.getWriting_list().get(i).getScrap_amount(),
                decode.getWriting_list().get(i).getPicture_url(),
                decode.getWriting_list().get(i).getWriting_name(),
                decode.getWriting_list().get(i).getExplanation()

        ));
    }
    adapter.notifyDataSetChanged();
}
@Override
public void onFailure(Call<KnowHowBean> call, Throwable t) {
    Log.e("failure", t.getMessage());
}
});
}
}
}

```

상위패키지	items	하위패키지
CommunityDetailBeanItem.java		
<pre> package sungkyul.ac.kr.leefrom.items; /** * Created by MiSeon on 2016-05-30. * json을 받아줄 클래스를 생성 * json의 키에 해당하는 값들을 변수 이름으로 지정 */ public class CommunityDetailBeanItem { String material_unique_key; String material_name; String material_explanation; String material_price; String subcontractor_name; String subcontractor_image_url; public String getMaterial_explanation() { return material_explanation; } public void setMaterial_explanation(String material_explanation) { this.material_explanation = material_explanation; } public String getMaterial_name() { return material_name; } public void setMaterial_name(String material_name) { </pre>		

```

        this.material_name = material_name;
    }
    public String getMaterial_price() {
        return material_price;
    }
    public void setMaterial_price(String material_price) {
        this.material_price = material_price;
    }
    public String getMaterial_unique_key() {
        return material_unique_key;
    }
    public void setMaterial_unique_key(String material_unique_key) {
        this.material_unique_key = material_unique_key;
    }
    public String getSubcontractor_image_url() {
        return subcontractor_image_url;
    }
    public void setSubcontractor_image_url(String subcontractor_image_url) {
        this.subcontractor_image_url = subcontractor_image_url;
    }
    public String getSubcontractor_name() {
        return subcontractor_name;
    }
    public void setSubcontractor_name(String subcontractor_name) {
        this.subcontractor_name = subcontractor_name;
    }
}

```

CommunityItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by MiSeon on 2016-05-20.
 */
public class CommunityItem {
    private int cNumber;
    private String cName;
    private String cCount;
    private String cContent;
    private String cImageURL;
    private String cTime;
    public String getTime() {
        return cTime;
    }
    public void setTime(String cTime) {
        this.cTime = cTime;
    }
    public int getcNumber() {

```

```

        return cNumber;
    }
    public void setcNumber(int cNumber) {
        this.cNumber = cNumber;
    }
    public String getName() {
        return cName;
    }
    public void setName(String cName) {
        this.cName = cName;
    }
    public String getCount() {
        return cCount;
    }
    public void setcCount(String cCount) {
        this.cCount = cCount;
    }
    public String getContent() {
        return cContent;
    }
    public void setcContent(String cContent) {
        this.cContent = cContent;
    }
    public String getImageURL() {
        return clmageURL;
    }
    public void setclmageURL(String clmageURL) {
        this.clmageURL = clmageURL;
    }
    public CommunityItem(int number, String name, String count, String content, String img, String
time) {
        cNumber = number;
        cName = name;
        cCount = count;
        cContent = content;
        clmageURL = img;
        cTime = time;
    }
}

```

CommunityListBeanItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by MiSeon on 2016-05-26.
 * json을 받아줄 클래스를 생성
 * json의 키에 해당하는 값들을 변수 이름으로 지정
 */

```

```

public class CommunityListBeanItem {
    private int community_unique_key;
    private String img;
    private String name;
    private String community_writing_contents;
    private String community_writing_date;
    private String reply_community_amount;
    public String getReply_community_amount() {
        return reply_community_amount;
    }
    public void setReply_community_amount(String reply_community_amount) {
        this.reply_community_amount = reply_community_amount;
    }
    public int getCommunity_unique_key() {
        return community_unique_key;
    }
    public void setCommunity_unique_key(int community_unique_key) {
        this.community_unique_key = community_unique_key;
    }
    public String getCommunity_writing_date() {
        return community_writing_date;
    }
    public void setCommunity_writing_date(String community_writing_date) {
        this.community_writing_date = community_writing_date;
    }
    public String getImg() {
        return img;
    }
    public void setImg(String img) {
        this.img = img;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getCommunity_writing_contents() {
        return community_writing_contents;
    }
    public void setCommunity_writing_contents(String community_writing_contents) {
        this.community_writing_contents = community_writing_contents;
    }
}

```

CommunityReplyBeanItem.java

```
package sungkyul.ac.kr.leeform.items;
```



```

/**
 * Created by MiSeon on 2016-05-29.
 * json을 받아줄 클래스를 생성
 * json의 키에 해당하는 값들을 변수 이름으로 지정
 */
public class CommunityReplyBeanItem {
    String name;
    String img;
    String reply_community_contents;
    String reply_date;
    public String getReply_date() {
        return reply_date;
    }
    public void setReply_date(String reply_date) {
        this.reply_date = reply_date;
    }
    public String getReply_community_contents() {
        return reply_community_contents;
    }
    public void setReply_community_contents(String reply_community_contents) {
        this.reply_community_contents = reply_community_contents;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getImg() {
        return img;
    }
    public void setImg(String img) {
        this.img = img;
    }
}

```

CreateKnowHowItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by HunJin on 2016-05-18.
 */
public class CreateKnowHowItem {
    private int number;
    private int mImg;
    private String imgUrl;
    private String explain;
    public int getNumber() {

```

```

        return number;
    }
    public int getmImg() {
        return mImg;
    }
    public String getExplain() {
        return explain;
    }
    public String getImgUrl() {
        return imgUrl;
    }
    public CreateKnowHowItem(int number, String mImg, String explain) {
        this.number = number;
        this.imgUrl = mImg;
        this.explain = explain;
    }
    public CreateKnowHowItem(int number, int mImg, String explain) {
        this.number = number;
        this.mImg = mImg;
        this.explain = explain;
    }
}

```

getAlarmStateBeanItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by HunJin on 2016-06-13.
 */
public class getAlarmStateBeanItem {
    int user_unique_key;
    int set_alarm;
    public int getUser_unique_key() {
        return user_unique_key;
    }
    public void setUser_unique_key(int user_unique_key) {
        this.user_unique_key = user_unique_key;
    }
    public int getSet_alarm() {
        return set_alarm;
    }
    public void setSet_alarm(int set_alarm) {
        this.set_alarm = set_alarm;
    }
}

```

KnowHowDetailBeanContentsItem.java

```

package sungkyul.ac.kr.leeform.items;

/**

```

* Created by HunJin on 2016-06-09.

*/

```
public class KnowHowDetailBeanContentsItem {
    String picture_url;
    String writing_name;
    String explanation;
    String writing_date;
    String img;
    String name;
    String level;
    String making_time;
    String price;
    String cost;
    String scrap_amount;
    public String getScrap_amount() {
        return scrap_amount;
    }
    public void setScrap_amount(String scrap_amount) {
        this.scrap_amount = scrap_amount;
    }
    public String getCost() {
        return cost;
    }
    public void setCost(String cost) {
        this.cost = cost;
    }
    public String getPicture_url() {
        return picture_url;
    }
    public void setPicture_url(String picture_url) {
        this.picture_url = picture_url;
    }
    public String getWriting_name() {
        return writing_name;
    }
    public void setWriting_name(String writing_name) {
        this.writing_name = writing_name;
    }
    public String getExplanation() {
        return explanation;
    }
    public void setExplanation(String explanation) {
        this.explanation = explanation;
    }
    public String getWriting_date() {
        return writing_date;
    }
}
```

```

    }
    public void setWriting_date(String writing_date) {
        this.writing_date = writing_date;
    }
    public String getImg() {
        return img;
    }
    public void setImg(String img) {
        this.img = img;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getLevel() {
        return level;
    }
    public void setLevel(String level) {
        this.level = level;
    }
    public String getMaking_time() {
        return making_time;
    }
    public void setMaking_time(String making_time) {
        this.making_time = making_time;
    }
    public String getPrice() {
        return price;
    }
    public void setPrice(String price) {
        this.price = price;
    }
}

```

KnowHowDetailBeanPictureItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by HunJin on 2016-06-09.
 */
public class KnowHowDetailBeanPictureItem {
    String picture_url;
    String writing_contents;
    public String getPicture_url() {
        return picture_url;
    }
}

```

```

    public void setPicture_url(String picture_url) {
        this.picture_url = picture_url;
    }
    public String getWriting_contents() {
        return writing_contents;
    }
    public void setWriting_contents(String writing_contents) {
        this.writing_contents = writing_contents;
    }
}

```

KnowHowWritingBeanItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by HunJin on 2016-05-28.
 * json을 받아줄 클래스를 생성
 * json의 키에 해당하는 값들을 변수 이름으로 지정
 */
public class KnowHowWritingBeanItem {
    String writing_unique_key;
    public String getWriting_unique_key() {
        return writing_unique_key;
    }
    public void setWriting_unique_key(String writing_unique_key) {
        this.writing_unique_key = writing_unique_key;
    }
}

```

LicenseBeanItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by HunJin on 2016-06-13.
 */
public class LicenseBeanItem {
    int license_unique_key;
    String license_writing_contents;
    String license_name;
    public int getLicense_unique_key() {
        return license_unique_key;
    }
    public void setLicense_unique_key(int license_unique_key) {
        this.license_unique_key = license_unique_key;
    }
    public String getLicense_writing_contents() {
        return license_writing_contents;
    }
    public void setLicense_writing_contents(String license_writing_contents) {
        this.license_writing_contents = license_writing_contents;
    }
}

```

```

    }
    public String getLicense_name() {
        return license_name;
    }
    public void setLicense_name(String license_name) {
        this.license_name = license_name;
    }
}

```

LicenseItem.java

```

package sungkyul.ac.kr.leeform.items;
/**
 * Created by HunJin on 2016-06-13.
 */
public class LicenseItem {
    private String licenseName;
    private int licenseUniqueKey;
    private String licenseContents;
    public String getLicenseName() {
        return licenseName;
    }
    public int getLicenseUniqueKey() {
        return licenseUniqueKey;
    }
    public String getLicenseContents() {
        return licenseContents;
    }
    public LicenseItem(int key, String name, String contents) {
        licenseUniqueKey = key;
        licenseName = name;
        licenseContents = contents;
    }
}

```

MainListItem.java

```

package sungkyul.ac.kr.leeform.items;
/**
 * Created by HunJin on 2016-05-10.
 */
public class MainListItem {
    private int mNumber;
    private String mCost;
    private String mTime;
    private String mLike;
    private String mName;
    private String mKeyword;
    private int mImg;
    private String mUrl;
}

```

```

public String getmKeyWord() {
    return mKeyWord;
}
public void setmKeyWord(String mKeyWord) {
    this.mKeyWord = mKeyWord;
}
public String getmName() {
    return mName;
}
public void setmName(String mName) {
    this.mName = mName;
}
public int getmNumber() {
    return mNumber;
}
public void setmNumber(int mNumber) {
    this.mNumber = mNumber;
}
public String getmCost() {
    return mCost;
}
public void setmCost(String mCost) {
    this.mCost = mCost;
}
public String getmTime() {
    return mTime;
}
public void setmTime(String mTime) {
    this.mTime = mTime;
}
public String getmLike() {
    return mLike;
}
public void setmLike(String mLike) {
    this.mLike = mLike;
}
public int getmImg() {
    return mImg;
}
public void setmImg(int mImg) {
    this.mImg = mImg;
}
public String getmUrl() {
    return mUrl;
}
public void setmUrl(String mUrl) {

```

```

        this.mUrl = mUrl;
    }

    public MainListItem(int number, String cost, String time, String like, String url, String name, String
    keyWord) {
        mNumber = number;
        mCost = cost;
        mTime = time;
        mLike = like;
        mUrl = url;
        mKeyWord = keyWord;
        mName = name;
    }
}

```

MaterialDetailBeanItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by user on 2016-06-12.
 */
public class MaterialDetailBeanItem {
    int material_unique_key;
    String material_name;
    String material_explanation;
    String material_price;
    String subcontractor_name;
    String subcontractor_image_url;
    public String getMaterial_explanation() {
        return material_explanation;
    }
    public void setMaterial_explanation(String material_explanation) {
        this.material_explanation = material_explanation;
    }
    public String getMaterial_name() {
        return material_name;
    }
    public void setMaterial_name(String material_name) {
        this.material_name = material_name;
    }
    public String getMaterial_price() {
        return material_price;
    }
    public void setMaterial_price(String material_price) {
        this.material_price = material_price;
    }
    public int getMaterial_unique_key() {
        return material_unique_key;
    }
}

```



```

    public void setMaterial_unique_key(int material_unique_key) {
        this.material_unique_key = material_unique_key;
    }
    public String getSubcontractor_image_url() {
        return subcontractor_image_url;
    }
    public void setSubcontractor_image_url(String subcontractor_image_url) {
        this.subcontractor_image_url = subcontractor_image_url;
    }
    public String getSubcontractor_name() {
        return subcontractor_name;
    }
    public void setSubcontractor_name(String subcontractor_name) {
        this.subcontractor_name = subcontractor_name;
    }
}

```

MaterialDetailBeanPictureItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by user on 2016-06-12.
 */
public class MaterialDetailBeanPictureItem {
    String material_picture_url;
    public String getMaterial_picture_url() {
        return material_picture_url;
    }
    public void setMaterial_picture_url(String material_picture_url) {
        this.material_picture_url = material_picture_url;
    }
}

```

MaterialGridItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by KyungHee on 2016-05-12.
 */
public class MaterialGridItem {
    private int mKey;
    private String mName;
    private String mUrl;
    private String mPrice;
    public MaterialGridItem(int material_unique_key, String material_picture_url, String material_name,
String material_price) {
        mKey = material_unique_key;
        mUrl = material_picture_url;
        mName = material_name;
        mPrice = material_price;
    }
}

```

```

    public int getmKey() {
        return mKey;
    }
    public void setmKey(int mKey) {
        this.mKey = mKey;
    }
    public String getmName() {
        return mName;
    }
    public void setmName(String mName) {
        this.mName = mName;
    }
    public String getmPrice() {
        return mPrice;
    }
    public void setmPrice(String mPrice) {
        this.mPrice = mPrice;
    }
    public String getmUrl() {
        return mUrl;
    }
    public void setmUrl(String mUrl) {
        this.mUrl = mUrl;
    }
}

```

MaterialListBeanItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by HunJin on 2016-06-12.
 */
public class MaterialListBeanItem {
    int material_unique_key;
    String material_name;
    String material_picture_url;
    String material_price;
    public int getMaterial_unique_key() {
        return material_unique_key;
    }
    public void setMaterial_unique_key(int material_unique_key) {
        this.material_unique_key = material_unique_key;
    }
    public String getMaterial_name() {
        return material_name;
    }
    public void setMaterial_name(String material_name) {
        this.material_name = material_name;
    }
}

```

```

    }

    public String getMaterial_picture_url() {
        return material_picture_url;
    }

    public void setMaterial_picture_url(String material_picture_url) {
        this.material_picture_url = material_picture_url;
    }

    public String getMaterial_price() {
        return material_price;
    }

    public void setMaterial_price(String material_price) {
        this.material_price = material_price;
    }
}

```

NoticeBeanItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by HunJin on 2016-06-13.
 */
public class NoticeBeanItem {
    int notice_unique_key;
    String notice_writing_contents;
    String notice_date;
    int notice_number;
    public int getNotice_unique_key() {
        return notice_unique_key;
    }
    public void setNotice_unique_key(int notice_unique_key) {
        this.notice_unique_key = notice_unique_key;
    }
    public String getNotice_writing_contents() {
        return notice_writing_contents;
    }
    public void setNotice_writing_contents(String notice_writing_contents) {
        this.notice_writing_contents = notice_writing_contents;
    }
    public String getNotice_date() {
        return notice_date;
    }
    public void setNotice_date(String notice_date) {
        this.notice_date = notice_date;
    }
    public int getNotice_number() {
        return notice_number;
    }
    public void setNotice_number(int notice_number) {

```

```

        this.notice_number = notice_number;
    }
}

```

Noticeltem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by HunJin on 2016-06-13.
 */
public class Noticeltem {
    private String name;
    private int nNoticeUniqueKey;
    private String date;
    private int number;
    public String getName() {
        return name;
    }
    public int getnNoticeUniqueKey() {
        return nNoticeUniqueKey;
    }
    public String getDate() {
        return date;
    }
    public int getNumber() {
        return number;
    }
    public Noticeltem(int noticeUniqueKey, String name, String date, int number) {
        this.name = name;
        this.nNoticeUniqueKey = noticeUniqueKey;
        this.date = date;
        this.number = number;
    }
}

```

Replyltem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by MiSeon on 2016-05-26.
 */
public class Replyltem {
    private int rNumber;
    private String rName;
    private String rContent;
    private String rImg;
    private String rTime;
    public String getrTime() {
        return rTime;
    }
}

```

```

    public void setrTime(String rTime) {
        this.rTime = rTime;
    }
    public int getrNumber() {
        return rNumber;
    }
    public void setrNumber(int rNumber) {
        this.rNumber = rNumber;
    }
    public String getrName() {
        return rName;
    }
    public void setrName(String rName) {
        this.rName = rName;
    }
    public String getrContent() {
        return rContent;
    }
    public void setrContent(String rContent) {
        this.rContent = rContent;
    }
    public String getrImg() {
        return rImg;
    }
    public void setrImg(String rImg) {
        this.rImg = rImg;
    }
    public ReplyItem(String name, String content, String img, String time) {
        rName = name;
        rContent = content;
        rImg = img;
        rTime = time;
    }
}

```

UserBeanItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by user on 2016-06-11.
 */
public class UserBeanItem {
    String name;
    String img;
    String authority;
    String address;
    String bank_name;
    String account_number;
}

```

```

String phone_number;
String account_name;
public String getName() {
    return name;
}
public void setName(String name) {
    this.name = name;
}
public String getImg() {
    return img;
}
public void setImg(String img) {
    this.img = img;
}
public String getAuthority() {
    return authority;
}
public void setAuthority(String authority) {
    this.authority = authority;
}
public String getAddress() {
    return address;
}
public void setAddress(String address) {
    this.address = address;
}
public String getBank_name() {
    return bank_name;
}
public void setBank_name(String bank_name) {
    this.bank_name = bank_name;
}
public String getAccount_number() {
    return account_number;
}
public void setAccount_number(String account_number) {
    this.account_number = account_number;
}
public String getPhone_number() {
    return phone_number;
}
public void setPhone_number(String phone_number) {
    this.phone_number = phone_number;
}
public String getAccount_name() {
    return account_name;
}

```

```

    }

    public void setAccount_name(String account_name) {
        this.account_name = account_name;
    }
}

```

UserInfoBeanItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by HunJin on 2016-06-02.
 * json을 받아줄 클래스를 생성
 * json의 키에 해당하는 값들을 변수 이름으로 지정
 */
public class UserInfoBeanItem {
    String user_unique_key;
    String kakao_unique_key;
    String img;
    String name;
    public String getUser_unique_key() {
        return user_unique_key;
    }
    public void setUser_unique_key(String user_unique_key) {
        this.user_unique_key = user_unique_key;
    }
    public String getKakao_unique_key() {
        return kakao_unique_key;
    }
    public void setKakao_unique_key(String kakao_unique_key) {
        this.kakao_unique_key = kakao_unique_key;
    }
    public String getImg() {
        return img;
    }
    public void setImg(String img) {
        this.img = img;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}

```

WritingBeanItem.java

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by HunJin on 2016-05-25.

```

```

* json을 받아줄 클래스를 생성
* json의 키에 해당하는 값들을 변수 이름으로 지정
*/
public class WritingBeanItem {
    String writing_unique_key;
    String writing_name;
    String price;
    String making_time;
    String picture_url;
    String scrap_amount;
    String explanation;
    String cost;
    public String getCost() {
        return cost;
    }
    public void setCost(String cost) {
        this.cost = cost;
    }
    public String getScrap_amount() {
        return scrap_amount;
    }
    public void setScrap_amount(String scrap_amount) {
        this.scrap_amount = scrap_amount;
    }
    public String getExplanation() {
        return explanation;
    }
    public void setExplanation(String explanation) {
        this.explanation = explanation;
    }
    public String getWriting_unique_key() {
        return writing_unique_key;
    }
    public void setWriting_unique_key(String writing_unique_key) {
        this.writing_unique_key = writing_unique_key;
    }
    public String getWriting_name() {
        return writing_name;
    }
    public void setWriting_name(String writing_name) {
        this.writing_name = writing_name;
    }
    public String getPrice() {
        return price;
    }
    public void setPrice(String price) {

```



```

        this.price = price;
    }
    public String getMaking_time() {
        return making_time;
    }
    public void setMaking_time(String making_time) {
        this.making_time = making_time;
    }
    public String getPicture_url() {
        return picture_url;
    }
    public void setPicture_url(String picture_url) {
        this.picture_url = picture_url;
    }
}

```

WritingDetailReplyListBeanItem

```

package sungkyul.ac.kr.leeform.items;

/**
 * Created by HunJin on 2016-06-13.
 */
public class WritingDetailReplyListBeanItem {
    String name;
    String img;
    String reply_writing_contents;
    String reply_date;
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getImg() {
        return img;
    }
    public void setImg(String img) {
        this.img = img;
    }
    public String getReply_writing_contents() {
        return reply_writing_contents;
    }
    public void setReply_writing_contents(String reply_writing_contents) {
        this.reply_writing_contents = reply_writing_contents;
    }
    public String getReply_date() {
        return reply_date;
    }
}

```

```

    public void setReply_date(String reply_date) {
        this.reply_date = reply_date;
    }
}

```

상위패키지	service	하위패키지
-------	---------	-------

MyGcmListenerService.java

```

package sungkyul.ac.kr.leeform.service;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.media.RingtoneManager;
import android.net.Uri;
import android.os.Bundle;
import android.support.v4.app.NotificationCompat;
import android.util.Log;
import com.google.android.gms.gcm.GcmListenerService;
import java.net.URLDecoder;
import sungkyul.ac.kr.leeform.MainActivity;
import sungkyul.ac.kr.leeform.activity.gcm.PopUpActivity;
/**
 * Created by HunJin on 2016-06-10.
 */
public class MyGcmListenerService extends GcmListenerService {
    private static final String TAG = "MyGcmListenerService";
    /**
     * Called when message is received.
     *
     * @param from SenderID of the sender.
     * @param data Data bundle containing message data as key/value pairs.
     * For Set of keys use data.keySet().
     */
    // [START receive_message]
    @Override
    public void onMessageReceived(String from, Bundle data) {
        try {
            String message = data.getString("ellord");
            message = URLDecoder.decode(message, "utf-8");
            Log.e(TAG, "From: " + from);
            Log.e(TAG, "Message: " + message);
            if (from.startsWith("/topics/")) {
                // message received from some topic.
            } else {
                // normal downstream message.
            }
        }
    }
}

```

```

// [START_EXCLUDE]
/**
 * Production applications would usually process the message here.
 * Eg: - Syncing with server.
 *      - Store message in local database.
 *      - Update UI.
 */
/**
 * In some cases it may be useful to show a notification indicating to the user
 * that a message was received.
 */
sendNotification(message);
getMessage(message);
} catch (Exception e) {
    e.printStackTrace();
}
// [END_EXCLUDE]
}
// [END receive_message]
/**
 * Create and show a simple notification containing the received GCM message.
 *
 * @param message GCM message received.
 */
private void sendNotification(String message) {
    Intent intent = new Intent(this, MainActivity.class);
    intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
    PendingIntent pendingIntent = PendingIntent.getActivity(this, 0 /* Request code */, intent,
        PendingIntent.FLAG_ONE_SHOT);

    Uri defaultSoundUri =
RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
    NotificationCompat.Builder notificationBuilder = new NotificationCompat.Builder(this)
        .setContentTitle("GCM Message")
        .setContentText(message)
        .setAutoCancel(true)
        .setSound(defaultSoundUri)
        .setContentIntent(pendingIntent);

    NotificationManager notificationManager =
        (NotificationManager) getSystemService(Context.NOTIFICATION_SERVICE);

    notificationManager.notify(0 /* ID of notification */, notificationBuilder.build());
}

private void getMessage(String message) {
    // 만약 팝업 알림 설정이켜져있으면 실행한다.
    // 팝업으로 사용할 액티비티를 호출할 인텐트를 작성한다.
    Intent popupIntent = new Intent(getApplicationContext(), PopUpActivity.class)
        .setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);

```

```

        popupIntent.putExtra("message", message);
        // 그리고 호출한다.
        getApplication().startActivity(popupIntent);
    }
}

```

MyInstanceDListenerService.java

```

package sungkyul.ac.kr.leeform.service;
import android.content.Intent;
import com.google.android.gms.iid.InstanceIDListenerService;
/**
 * Created by HunJin on 2016-06-10.
 */
public class MyInstanceIDListenerService extends InstanceIDListenerService {
    private static final String TAG = "MyInstanceIDLS";
    /**
     * Called if InstanceID token is updated. This may occur if the security of
     * the previous token had been compromised. This call is initiated by the
     * InstanceID provider.
     */
    // [START refresh_token]
    @Override
    public void onTokenRefresh() {
        // Fetch updated Instance ID token and notify our app's server of any changes (if applicable).
        Intent intent = new Intent(this, RegistrationIntentService.class);
        startService(intent);
    }
}

```

RegistrationIntentService.java

```

package sungkyul.ac.kr.leeform.service;
import android.app.IntentService;
import android.content.Intent;
import android.content.SharedPreferences;
import android.preference.PreferenceManager;
import android.support.v4.content.LocalBroadcastManager;
import android.util.Log;
import com.google.android.gms.gcm.GcmPubSub;
import com.google.android.gms.gcm.GoogleCloudMessaging;
import com.google.android.gms.iid.InstanceID;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;
import retrofit2.Call;
import retrofit2.Callback;
import retrofit2.Response;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;

```

```

import sungkyul.ac.kr.leeform.R;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.OnlyErrBean;
import sungkyul.ac.kr.leeform.utils.QuickstartPreferences;
import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo;
import sungkyul.ac.kr.leeform.utils.StaticURL;

/**
 * Created by HunJin on 2016-06-10.
 */
public class RegistrationIntentService extends IntentService {
    private static final String TAG = "RegIntentService";
    private static final String[] TOPICS = {"global"};
    public RegistrationIntentService() {
        super(TAG);
    }
    @Override
    protected void onHandleIntent(Intent intent) {
        SharedPreferences sharedPreferences =
        PreferenceManager.getDefaultSharedPreferences(this);
        try {
            // [START register_for_gcm]
            // Initially this call goes out to the network to retrieve the token, subsequent calls
            // are local.
            // R.string.gcm_defaultSenderId (the Sender ID) is typically derived from
            google-services.json.
            // See https://developers.google.com/cloud-messaging/android/start for details on this file.
            // [START get_token]
            InstanceID instanceID = InstanceID.getInstance(this);
            String token = instanceID.getToken(getString(R.string.gcm_defaultSenderId),
                GoogleCloudMessaging.INSTANCE_ID_SCOPE, null);
            // [END get_token]
            Log.i(TAG, "GCM Registration Token: " + token);
            // TODO: Implement this method to send any registration to your app's servers.
            sendRegistrationToServer(token);
            // Subscribe to topic channels
            subscribeTopics(token);
            // You should store a boolean that indicates whether the generated token has been
            // sent to your server. If the boolean is false, send the token to your server,
            // otherwise your server should have already received the token.

            sharedPreferences.edit().putBoolean(QuickstartPreferences.SENT_TOKEN_TO_SERVER,
                true).apply();
            // [END register_for_gcm]
        } catch (Exception e) {
            Log.d(TAG, "Failed to complete token refresh", e);
            // If an exception happens while fetching the new token or updating our registration data
            // on a third-party server, this ensures that we'll attempt the update at a later time.

```

```

sharedPreferences.edit().putBoolean(QuickstartPreferences.SENT_TOKEN_TO_SERVER,
false).apply();
    }
    // Notify UI that registration has completed, so the progress indicator can be hidden.
    Intent registrationComplete = new
Intent(QuickstartPreferences.REGISTRATION_COMPLETE);
    LocalBroadcastManager.getInstance(this).sendBroadcast(registrationComplete);
}
/**
 * Persist registration to third-party servers.
 *
 * Modify this method to associate the user's GCM registration token with any server-side account
 * maintained by your application.
 *
 * @param token The new token.
 */
private void sendRegistrationToServer(String token) {
    // Add custom implementation, as needed.
    setGCMTOKEN(token);
}
/**
 * Subscribe to any GCM topics of interest, as defined by the TOPICS constant.
 *
 * @param token GCM token
 * @throws IOException if unable to reach the GCM PubSub service
 */
// [START subscribe_topics]
private void subscribeTopics(String token) throws IOException {
    GcmPubSub pubSub = GcmPubSub.getInstance(this);
    for (String topic : TOPICS) {
        pubSub.subscribe(token, "/topics/" + topic, null);
    }
}
// [END subscribe_topics]
private void setGCMTOKEN(String token) {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(StaticURL.BASE_URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();

    Map<String, String> map = new HashMap<>();
    Log.e("user key
get", SaveDataMemberInfo.getAppPreferences(getApplicationContext(), "user_key"));
    map.put("user_unique_key",
SaveDataMemberInfo.getAppPreferences(getApplicationContext(), "user_key"));
    map.put("gcm", token);
    ConnectService connectService = retrofit.create(ConnectService.class);

```

```

Call<OnlyErrBean> call = connectService.setGCMTToken(map);
call.enqueue(new Callback<OnlyErrBean>() {
    @Override
    public void onResponse(Call<OnlyErrBean> call, Response<OnlyErrBean> response) {
        OnlyErrBean decode = response.body();
        if(decode.getErr().equals("0")) {
            Log.e("gcm token save","success");
        } else {
            Log.e("gcm token save","fail");
        }
    }
    @Override
    public void onFailure(Call<OnlyErrBean> call, Throwable t) {
    }
});
}
}

```

상위패키지	utils	하위패키지
BackPressCloseHandler.java		
<pre> package sungkyul.ac.kr.leeform.utils; import android.app.Activity; /** * Created by YongHoon on 2016-05-19. */ public class BackPressCloseHandler { private long backKeyPressedTime = 0; private Activity activity; public BackPressCloseHandler(Activity context) { this.activity = context; } public void onBackPressed() { if (System.currentTimeMillis() > backKeyPressedTime + 2000) { backKeyPressedTime = System.currentTimeMillis(); return; } if (System.currentTimeMillis() <= backKeyPressedTime + 2000) { activity.finish(); } } } </pre>		
DataProvider.java		
<pre> package sungkyul.ac.kr.leeform.utils; import java.util.HashMap; /** * Created by YongHoon on 2016-05-21. </pre>		

```

*/
public class DataProvider {
    // StaticURL 파싱을 하기 위해서는 이 부분을 사용하세요.
    HashMap<String, String> file_maps;
    // 이미지로부터 가져오기 위해서는 이 부분을 사용하세요.
    HashMap<String, Integer> file_maps_src;
    public DataProvider() {
        file_maps = new HashMap<>();
        file_maps_src = new HashMap<>();
    }
    public HashMap<String, String> getFileSrcHorizontal() {
        return file_maps;
    }
    public HashMap<String, Integer> getFileSrc() {
        return file_maps_src;
    }
    public void setHashFile(String fileNmae, String url) {
        file_maps.put(fileNmae, url);
    }
    public void setHashFile(String fileNmae, int url) {
        file_maps_src.put(fileNmae, url);
    }
}

```

DownloadImageTask.java

```

package sungkyul.ac.kr.leeform.utils;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.AsyncTask;
import android.util.Log;
import android.widget.ImageView;
import java.io.InputStream;
/**
 * Created by KyungHee on 2016-05-28.
 */
public class DownloadImageTask extends AsyncTask<String, Void, Bitmap> {
    ImageView bmlImage;
    public DownloadImageTask(ImageView bmlImage) {
        this.bmlImage = bmlImage;
    }
    protected Bitmap doInBackground(String... urls) {
        String urldisplay = urls[0];
        Bitmap mIcon11 = null;
        try {
            InputStream in = new java.net.URL(urldisplay).openStream();
            mIcon11 = BitmapFactory.decodeStream(in);
        } catch (Exception e) {

```



```

        Log.e("Error", e.getMessage());
        e.printStackTrace();
    }
    return mlcon11;
}
protected void onPostExecute(Bitmap result) {
    bmlImage.setImageBitmap(result);
}
}

```

EndString.java

```

package sungkyul.ac.kr.leeform.utils;
/**
 * Created by HunJin on 2016-06-13.
 */
public class EndString {
    public static String endString(String contents, int n) {
        if(contents.length()>n) {
            return contents.substring(0,n) + "...";
        } else {
            return contents;
        }
    }
}

```

GlobalApplication.java

```

package sungkyul.ac.kr.leeform.utils;
import android.app.Activity;
import android.app.Application;
import android.graphics.Bitmap;
import android.support.v4.util.LruCache;
import android.view.Display;
import com.android.volley.RequestQueue;
import com.android.volley.toolbox.ImageLoader;
import com.android.volley.toolbox.Volley;
import com.kakao.auth.KakaoSDK;
import sungkyul.ac.kr.leeform.adapter.KakaoSDKAdapter;
/**
 * Created by HunJin on 2016-05-26.
 * 카카오 로그인 시 필요한 클래스
 */
public class GlobalApplication extends Application {
    private static volatile GlobalApplication instance = null;
    private static volatile Activity currentActivity = null;
    private ImageLoader imageLoader;
    public static Activity getCurrentActivity() {
        return currentActivity;
    }
}

```

```

public static void setCurrentActivity(Activity currentActivity) {
    GlobalApplication.currentActivity = currentActivity;
}

/**
 * singleton 애플리케이션 객체를 얻는다.
 *
 * @return singleton 애플리케이션 객체
 */
public static GlobalApplication getGlobalApplicationContext() {
    if (instance == null)
        throw new IllegalStateException("this application does not inherit
com.kakao.GlobalApplication");
    return instance;
}

/**
 * 이미지 로더, 이미지 캐시, 요청 큐를 초기화한다.
 */
@Override
public void onCreate() {
    super.onCreate();
    instance = this;
    KakaoSDK.init(new KakaoSDKAdapter());
    final RequestQueue requestQueue = Volley.newRequestQueue(this);
    ImageLoader.ImageCache imageCache = new ImageLoader.ImageCache() {
        final LruCache<String, Bitmap> imageCache = new LruCache<String, Bitmap>(3);
        @Override
        public void putBitmap(String key, Bitmap value) {
            imageCache.put(key, value);
        }
        @Override
        public Bitmap getBitmap(String key) {
            return imageCache.get(key);
        }
    };
    imageLoader = new ImageLoader(requestQueue, imageCache);
}

/**
 * 이미지 로더를 반환한다.
 *
 * @return 이미지 로더
 */
public ImageLoader getImageLoader() {
    return imageLoader;
}

/**
 * 애플리케이션 종료시 singleton 어플리케이션 객체 초기화한다.

```

```

*/
@Override
public void onTerminate() {
    super.onTerminate();
    instance = null;
}
public static Display mDisplay;
public static void setDisplay(Display display) {
    mDisplay = display;
}
public static int getDisplayWidth() {
    return mDisplay.getWidth();
}
public static int getDisplayHeight() {
    return mDisplay.getHeight();
}
public int resize_Height(int width, int height, int resize_width) {
    return (this.getDisplayHeight() * resize_width) / getDisplayWidth();
}
}

```

LoadActivityList.java

```

package sungkyul.ac.kr.leeform.utils;
import android.app.Activity;
import java.util.ArrayList;
/**
 * Created by YongHoon on 2016-05-22.
 */
public class LoadActivityList {
    //onCreate되는 액티비티들 저장
    public static ArrayList<Activity> actList = new ArrayList<Activity>();
    // 저장한 모든 액티비티 종료
    public void closeActivity() {
        for (int i = 0; i < actList.size(); i++) {
            actList.get(i).finish();
        }
    }
}

```

NumZeroForm.java

```

package sungkyul.ac.kr.leeform.utils;
import android.content.Context;
import android.widget.TextView;
import com.hkm.slider.Indicators.NumContainer;
import sungkyul.ac.kr.leeform.R;
/**
 * Created by YongHoon on 2016-05-21.
 */

```

```

public class NumZeroForm extends NumContainer<TextView> {
    public NumZeroForm(Context c) {
        super(c, R.layout.numfield);
    }
}

```

QuickstarPreferences.java

```

package sungkyul.ac.kr.leeform.utils;
/**
 * Created by HunJin on 2016-06-10.
 */
public class QuickstartPreferences {
    public static final String SENT_TOKEN_TO_SERVER = "sentTokenToServer";
    public static final String REGISTRATION_COMPLETE = "registrationComplete";
}

```

RoundImageView.java

```

package sungkyul.ac.kr.leeform.utils;
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.PorterDuff;
import android.graphics.PorterDuffXfermode;
import android.graphics.Rect;
import android.graphics.drawable.BitmapDrawable;
import android.graphics.drawable.Drawable;
import android.util.AttributeSet;
import android.widget.ImageView;
/**
 * Created by MiSeon on 2016-05-20.
 */
public class RoundImageView extends ImageView {
    public RoundImageView(Context context) {
        super(context);
    }
    public RoundImageView(Context context, AttributeSet attrs) {
        super(context, attrs);
    }
    public RoundImageView(Context context, AttributeSet attrs, int defStyle) {
        super(context, attrs, defStyle);
    }
    @Override
    protected void onDraw(Canvas canvas) {
        Drawable drawable = getDrawable();
        if (drawable == null) {
            return;

```

```

    }
    if (getWidth() == 0 || getHeight() == 0) {
        return;
    }
    Bitmap mB = ((BitmapDrawable) drawable).getBitmap();
    Bitmap mBitmap = mB.copy(Bitmap.Config.ARGB_8888, true);
    int mWidth = getWidth();
    int mHeight = getHeight();
    Bitmap mRoundBitmap = getCroppedBitmap(mBitmap, mWidth);
    canvas.drawBitmap(mRoundBitmap, 0, 0, null);
}

public static Bitmap getCroppedBitmap(Bitmap bmp, int radius) {
    Bitmap mSmallBitmap;
    if (bmp.getWidth() != radius || bmp.getHeight() != radius) {
        float fSmallest = Math.min(bmp.getWidth(), bmp.getHeight());
        float fFactor = fSmallest / radius;
        mSmallBitmap = Bitmap.createScaledBitmap(bmp, (int) (bmp.getWidth() / fFactor), (int) (bmp.getHeight() / fFactor), false);
    } else {
        mSmallBitmap = bmp;
    }

    Bitmap mOutput = Bitmap.createBitmap(radius, radius, Bitmap.Config.ARGB_8888);
    Canvas mCanvas = new Canvas(mOutput);
    final int mColor = 0xffa19774;
    final Paint mPaint = new Paint();
    final Rect mRect = new Rect(0, 0, radius, radius);
    mPaint.setAntiAlias(true);
    mPaint.setFilterBitmap(true);
    mPaint.setDither(true);
    mCanvas.drawARGB(0, 0, 0, 0);
    mPaint.setColor(Color.parseColor("#BAB399"));
    mCanvas.drawCircle(radius / 2 + 0.7f, radius / 2 + 0.7f, radius / 2 + 0.1f, mPaint);
    mPaint.setXfermode(new PorterDuffXfermode(PorterDuff.Mode.SRC_IN));
    mCanvas.drawBitmap(mSmallBitmap, mRect, mRect, mPaint);
    return mOutput;
}
}

```

SaveData.java

```

package sungkyul.ac.kr.leeform.utils;
import android.content.Context;
import android.content.SharedPreferences;
/**
 * Created by YongHoon on 2016-05-26.
 */
public class SaveData {
    //파일에서 데이터 가져오는거
}

```

```

    public static String getAppPreferences(Context context, String key) {
        String returnValue = null;
        SharedPreferences pref = null;
        pref = context.getSharedPreferences("leeform", 0);
        returnValue = pref.getString(key, "");
        return returnValue;
    }
    //파일에 저장하는거
    public static void setAppPreferences(Context context, String key, String value) {
        SharedPreferences pref = null;
        pref = context.getSharedPreferences("leeform", 0);
        SharedPreferences.Editor prefEditor = pref.edit();
        prefEditor.putString(key, value);
        prefEditor.commit();
    }
}

```

SaveDataMemberInfo.java

```

package sungkyul.ac.kr.leeform.utils;
import android.content.Context;
import android.content.SharedPreferences;
/**
 * Created by YongHoon on 2016-05-28.
 */
public class SaveDataMemberInfo {
    public static void setAppPreferences
        (Context context, String key, String value) {
        SharedPreferences pref = null;
        pref = context.getSharedPreferences("logo", 0);
        SharedPreferences.Editor prefEditor = pref.edit();
        prefEditor.putString(key, value);
        prefEditor.commit();
    }
    public static String getAppPreferences
        (Context context, String key) {
        String returnValue = null;
        SharedPreferences pref = null;
        pref = context.getSharedPreferences("logo", 0);
        returnValue = pref.getString(key, "");
        return returnValue;
    }
}

```

StaticURL.java

```

package sungkyul.ac.kr.leeform.utils;
/**
 * Created by KyungHee on 2016-05-24.
 */

```

```

public class StaticURL {
    final static public String BASE_URL = "http://14.63.196.255/api/";
    final static public String IMAGE_URL = "http://14.63.196.255/image/";
    final static public String IMAGE_UPLOAD_URL = "http://14.63.196.255/api/upload.php";
}

```

TimeMaximum.java

```

package sungkyul.ac.kr.leeform.utils;

/**
 * Created by HunJin on 2016-06-06.
 */
public class TimeMaximum {
    public static final int SEC = 60;
    public static final int MIN = 60;
    public static final int HOUR = 24;
    public static final int DAY = 30;
    public static final int MONTH = 12;
}

```

TimeTransForm.java

```

package sungkyul.ac.kr.leeform.utils;
import java.text.ParseException;

/**
 * Created by HunJin on 2016-06-06.
 */
public class TimeTransForm {
    public static String formatTimeString(String str) throws ParseException {
        java.text.SimpleDateFormat format = new java.text.SimpleDateFormat(
            "yyyy-MM-dd HH:mm:ss");
        java.util.Date date = format.parse(str);
        long curTime = System.currentTimeMillis();
        long regTime = date.getTime();
        long diffTime = (curTime - regTime) / 1000;
        String msg = null;
        if (diffTime < TimeMaximum.SEC) {
            msg = "방금 전";
        } else if ((diffTime /= TimeMaximum.SEC) < TimeMaximum.MIN) {
            msg = diffTime + "분 전";
        } else if ((diffTime /= TimeMaximum.MIN) < TimeMaximum.HOUR) {
            msg = (diffTime) + "시간 전";
        } else if ((diffTime /= TimeMaximum.HOUR) < TimeMaximum.DAY) {
            msg = (diffTime) + "일 전";
        } else if ((diffTime /= TimeMaximum.DAY) < TimeMaximum.MONTH) {
            msg = (diffTime) + "달 전";
        } else {
            msg = str;
        }
        return msg;
    }
}

```

```

    }
}

```

상위패키지	leeform	하위패키지
	MainActivity.java	
	<pre> package sungkyul.ac.kr.leeform; import android.content.BroadcastReceiver; import android.content.Context; import android.content.Intent; import android.content.IntentFilter; import android.content.SharedPreferences; import android.graphics.Bitmap; import android.graphics.BitmapFactory; import android.os.Bundle; import android.os.Handler; import android.preference.PreferenceManager; import android.support.design.widget.TabLayout; import android.support.v4.content.LocalBroadcastManager; import android.support.v4.view.ViewPager; import android.support.v7.app.AppCompatActivity; import android.support.v7.widget.Toolbar; import android.util.Log; import android.view.View; import android.widget.AdapterView; import android.widget.AdapterView; import android.widget.ArrayAdapter; import android.widget.ImageView; import android.widget.ListView; import android.widget.TextView; import android.widget.Toast; import com.google.android.gms.common.ConnectionResult; import com.google.android.gms.common.GoogleApiAvailability; import com.navdrawer.SimpleSideDrawer; import java.io.InputStream; import java.net.URL; import java.util.HashMap; import java.util.Map; import retrofit2.Call; import retrofit2.Callback; import retrofit2.Response; import retrofit2.Retrofit; import retrofit2.converter.gson.GsonConverterFactory; import sungkyul.ac.kr.leeform.activity.navigation.MyPageActivity; import sungkyul.ac.kr.leeform.activity.navigation.PurchaseListActivity; import sungkyul.ac.kr.leeform.activity.navigation.RegistSellerActivity; import sungkyul.ac.kr.leeform.activity.navigation.SaleListActivity; import sungkyul.ac.kr.leeform.activity.search.KnowHowSearchActivity; </pre>	


```

import sungkyul.ac.kr.leeform.activity.search.MaterialSearchActivity;
import sungkyul.ac.kr.leeform.activity.settings.SettingActivity;
import sungkyul.ac.kr.leeform.adapter.MainFragmentAdapter;
import sungkyul.ac.kr.leeform.dao.ConnectService;
import sungkyul.ac.kr.leeform.dto.UserBean;
import sungkyul.ac.kr.leeform.dto.UserInfoBean;
import sungkyul.ac.kr.leeform.service.RegistrationIntentService;
import sungkyul.ac.kr.leeform.utils.BackPressCloseHandler;
import sungkyul.ac.kr.leeform.utils.LoadActivityList;
import sungkyul.ac.kr.leeform.utils.QuickstartPreferences;
import sungkyul.ac.kr.leeform.utils.SaveData;
import sungkyul.ac.kr.leeform.utils.SaveDataMemberInfo;
import sungkyul.ac.kr.leeform.utils.StaticURL;
/**
 * Created by HunJin on 2016-05-09.
 * <p>
 * 애플리케이션의 기본이 되는 메인 액티비티입니다.
 */
public class MainActivity extends AppCompatActivity {
    ViewPager viewPager;
    private static final int PLAY_SERVICES_RESOLUTION_REQUEST = 9000;
    private static final String TAG = "MainActivity";
    private BroadcastReceiver mRegistrationBroadcastReceiver;
    private boolean isReceiverRegistered;
    private static String URL = StaticURL.BASE_URL;
    private BackPressCloseHandler backPressCloseHandler;
    private SimpleSideDrawer slidingMenu;
    private TabLayout tabLayout;
    private ListView lstNavItem;
    private ImageView imgNavUser;
    private TextView txtNavUserNickName;
    private long userId;
    private String userUniqueKey;
    private String userNickName, userImagePath;
    private String userNickNameIn, userImagePathIn;
    private Toolbar toolbar;
    String[] item;
    Handler handler = new Handler();
    /**
     * getIntent 값
     * UserId : 유저 식별 키
     * NickName : 카카오 사용자 닉네임
     * Image : 유저 썸네일 이미지
     *
     * @param savedInstanceState
     */
}

```

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    // 로그아웃 할 때 열려있는 액티비티 모두 닫기 위해 리스트에 저장
    LoadActivityList.actList.add(MainActivity.this);
    Intent it = getIntent();
    userId = it.getExtras().getLong("UserId");
    userNickName = it.getExtras().getString("NickName");
    userImagePath = it.getExtras().getString("Image");
    tabInitialization();
    initializeLayout();
    getAuthority();
    setListener();
    checkUser();
    gcm();
}

private void gcm() {
    mRegistrationBroadcastReceiver = new BroadcastReceiver() {
        @Override
        public void onReceive(Context context, Intent intent) {
            SharedPreferences sharedPreferences =
                PreferenceManager.getDefaultSharedPreferences(context);
            boolean sentToken = sharedPreferences
                .getBoolean(QuickstartPreferences.SENT_TOKEN_TO_SERVER, false);
            if (sentToken) {
                Log.e("sentToken", R.string.gcm_send_message + "");
            } else {
                Log.e("information", R.string.token_error_message + "");
            }
        }
    };
    registerReceiver();
    if (checkPlayServices()) {
        // Start IntentService to register this application with GCM.
        Intent intent = new Intent(this, RegistrationIntentService.class);
        startService(intent);
    }
}

/**
 * 회원의 기본적인 정보를 저장하는 메서드입니다.
 * 첫 로그인일 경우 디비에 저장을 하고,
 * 이후에는 디비로부터 데이터를 가져옵니다.
 *
 * @param userId
 * @param userNickName

```

```

    * @param imagePath
    */
    private void setUser(final long userId, String userNickName, String imagePath) {
        Retrofit retrofit = new Retrofit.Builder()
            .baseUrl(URL)
            .addConverterFactory(GsonConverterFactory.create())
            .build();

        Map<String, String> data = new HashMap<>();
        data.put("kakao_unique_key", userId + "");
        data.put("img", imagePath);
        data.put("name", userNickName);
        ConnectService connectService = retrofit.create(ConnectService.class);
        Call<UserInfoBean> call = connectService.setUserInfo(data);
        call.enqueue(new Callback<UserInfoBean>() {
            @Override
            public void onResponse(Call<UserInfoBean> call, Response<UserInfoBean> response)
        {
            Log.e("setUser", response.code() + "");
            checkUser();
        }
            @Override
            public void onFailure(Call<UserInfoBean> call, Throwable t) {
                Log.e("failure", t.getMessage());
            }
        });
    }
    /**
     * 회원이 등록이 되어있는지 체크를 합니다.
     * 등록이 되어있을 경우 회원정보를 가져오며,
     * 등록이 되지 않은 경우 등록을 하는
     * setUser() 메서드를 호출합니다.
     */
    private void checkUser() {
        Retrofit retrofit = new Retrofit.Builder()
            .baseUrl(URL)
            .addConverterFactory(GsonConverterFactory.create())
            .build();

        ConnectService connectService = retrofit.create(ConnectService.class);
        Call<UserInfoBean> call = connectService.getUserInfo(userId + "");
        call.enqueue(new Callback<UserInfoBean>() {
            @Override
            public void onResponse(Call<UserInfoBean> call, Response<UserInfoBean> response)
        {
            UserInfoBean decode = response.body();
            Log.e("err", decode.getErr());
            String err = decode.getErr();

```

```

        if (err.equals("0")) {
            userUniqueKey = decode.getKakao_user_info().get(0).getUser_unique_key();
            userNickNameIn = decode.getKakao_user_info().get(0).getName();
            userImagePathIn = decode.getKakao_user_info().get(0).getImg();
            SaveDataMemberInfo.setAppPreferences(getApplicationContext(), "user_key",
userUniqueKey);
            navigationSetting();
        } else if (err.equals("4")) {
            setUser(userId, userNickName, userImagePath);
        }
    }
    @Override
    public void onFailure(Call<UserInfoBean> call, Throwable t) {
        Log.e("failure", t.getMessage());
    }
});
}
/**
 * 네비게이션 세팅 (사용자 이미지, 닉네임)
 */
private void navigationSetting() {
    txtNavUserNickName.setText(userNickNameIn);
    if (userImagePath != null) {
        userImageSetting();
    }
}
/**
 * 스레드를 사용하여 이미지 가져오기
 */
private void userImageSetting() {
    Thread thread = new Thread(new Runnable() {
        @Override
        public void run() {
            try {
                URL url = new URL(userImagePath);
                InputStream inputStream = url.openStream();
                final Bitmap bm = BitmapFactory.decodeStream(inputStream);
                handler.post(new Runnable() {
                    @Override
                    public void run() {
                        imgNavUser.setImageBitmap(bm);
                    }
                });
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

```

```

    }
    });
    thread.start();
}
/**
 * 화면에 보여줄 정보 초기화
 */
private void initializeLayout() {
    slidingMenu = new SimpleSideDrawer(MainActivity.this);
    slidingMenu.setLeftBehindContentView(R.layout.nav_view);
    lstNavItem = (ListView) slidingMenu.findViewById(R.id.lstNavItem);
    // 취소버튼 눌렀을 때 핸들러
    backPressCloseHandler = new BackPressCloseHandler(this);
    // 네비게이션 정보 설정
    imgNavUser = (ImageView) findViewById(R.id.imgNavUser);
    txtNavUserNickName = (TextView) findViewById(R.id.txtNavUserNickName);
    toolbar = (Toolbar) findViewById(R.id.toolbar);
    toolbar.setContentInsetsAbsolute(0, 0);
}
/**
 * 리스너 설정
 */
private void setListener() {
    ImageView imgNav = (ImageView) findViewById(R.id.imgNav);
    imgNav.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            slidingMenu.toggleLeftDrawer();
        }
    });
    ImageView imgSearch = (ImageView) findViewById(R.id.imgSearch);
    imgSearch.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            if (tabLayout.getSelectedTabPosition() == 0) {
                Intent it = new Intent(getApplicationContext(), KnowHowSearchActivity.class);
                startActivity(it);
                overridePendingTransition(R.anim.common_slide_from_right,
                    R.anim.common_slide_to_left);
            } else if (tabLayout.getSelectedTabPosition() == 1) {
                startActivity(new Intent(getApplicationContext(), MaterialSearchActivity.class));
                overridePendingTransition(R.anim.common_slide_from_right,
                    R.anim.common_slide_to_left);
            } else {
                Toast.makeText(getApplicationContext(), "커뮤니티에선 지원하지 않는
기능입니다.", Toast.LENGTH_SHORT).show();
            }
        }
    });
}

```

```

    }
}
});
item = getResources().getStringArray(R.array.nav);
if (SaveData.getAppPreferences(getApplicationContext(), "isAuthority").equals("1")) {
    item[2] = "판매 내역";
}else{
    item[2] = "판매자 등록";
}
lstNavItem = (ListView) slidingMenu.findViewById(R.id.lstNavItem);
ArrayAdapter<String> yada = new ArrayAdapter<String>(this, R.layout.nav_item, item);
lstNavItem.setAdapter(yada);
lstNavItem.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {
        switch (i) {
            // 내 정보
            case 0:
                Intent intent = new Intent(getApplicationContext(), MyPageActivity.class);
                startActivity(intent);
                break;
            // 구매내역
            case 1:
                Intent itPurchaseList = new Intent(getApplicationContext(),
PurchaseListActivity.class);
                startActivity(itPurchaseList);
                break;
            // 판매자 등록
            case 2:
                // 판매자 등록이 된상태면 판매내역으로
                if (SaveData.getAppPreferences(getApplicationContext(),
"isAuthority").equals("1")) {
                    Intent itSetting = new Intent(getApplicationContext(),
SaleListActivity.class);
                    startActivity(itSetting);
                }
                // 판매자 등록이 안되어 있으면 판매자 등록으로
                else {
                    Intent itRegistSeller = new Intent(getApplicationContext(),
RegistSellerActivity.class);
                    startActivity(itRegistSeller);
                }
                break;
            // 설정
            case 3:
                Intent itSetting = new Intent(getApplicationContext(), SettingActivity.class);
                startActivity(itSetting);

```

```

        break;
    }
    slidingMenu.closeLeftSide();
}
});
}
private void getAuthority() {
    Retrofit retrofit = new Retrofit.Builder()
        .baseUrl(URL)
        .addConverterFactory(GsonConverterFactory.create())
        .build();
    ConnectService connectService = retrofit.create(ConnectService.class);
    String userKey = SaveDataMemberInfo.getAppPreferences(getApplicationContext(),
"user_key");
    final Call<UserBean> call = connectService.getUserDetail(userKey);
    call.enqueue(new Callback<UserBean>() {
        @Override
        public void onResponse(Call<UserBean> call, Response<UserBean> response) {
            UserBean decode = response.body();
            String authority = decode.getMyinfo_detail().get(0).getAuthority();
            SaveData.setAppPreferences(getApplicationContext(), "isAuthority", authority);
        }
        @Override
        public void onFailure(Call<UserBean> call, Throwable t) {
            Log.e("failure", t.getMessage());
        }
    });
}
/**
 * 뒤로가기 키를 눌렀을 때
 */
@Override
public void onBackPressed() {
    // 네비게이션이 열려있으면
    if (!slidingMenu.isClosed()) {
        slidingMenu.closeLeftSide();
        return;
    }
    //핸들러 작동
    backPressCloseHandler.onBackPressed();
    Toast.makeText(getApplicationContext(), "한 번 더 누르면 앱이 종료됩니다",
    Toast.LENGTH_SHORT).show();
}
/**
 * 탭 뷰 초기화
 */

```

```

private void tabInitialization() {
    viewPager = (ViewPager) findViewById(R.id.mainViewPager);
    viewPager.setOffscreenPageLimit(2);
    MainFragmentAdapter mainFragmentAdapter = new
MainFragmentAdapter(getSupportFragmentManager(), MainActivity.this);
    viewPager.setAdapter(mainFragmentAdapter);
    tabLayout = (TabLayout) findViewById(R.id.mainTab);
    tabLayout.setupWithViewPager(viewPager);
}

/**
 * Check the device to make sure it has the Google Play Services APK. If
 * it doesn't, display a dialog that allows users to download the APK from
 * the Google Play Store or enable it in the device's system settings.
 */
private boolean checkPlayServices() {
    GoogleApiAvailability apiAvailability = GoogleApiAvailability.getInstance();
    int resultCode = apiAvailability.isGooglePlayServicesAvailable(this);
    if (resultCode != ConnectionResult.SUCCESS) {
        if (apiAvailability.isUserResolvableError(resultCode)) {
            apiAvailability.getErrorDialog(this, resultCode,
PLAY_SERVICES_RESOLUTION_REQUEST)
                .show();
        } else {
            Log.i(TAG, "This device is not supported.");
            finish();
        }
        return false;
    }
    return true;
}

private void registerReceiver() {
    if (!isReceiverRegistered) {

LocalBroadcastManager.getInstance(this).registerReceiver(mRegistrationBroadcastReceiver,
        new IntentFilter(QuickstartPreferences.REGISTRATION_COMPLETE));
        isReceiverRegistered = true;
    }
}

@Override
protected void onPause() {

LocalBroadcastManager.getInstance(this).unregisterReceiver(mRegistrationBroadcastReceiver);
    isReceiverRegistered = false;
    super.onPause();
}

@Override
protected void onResume() {

```



```
        super.onResume();  
        registerReceiver();  
        setListener();  
    }  
}
```