프로젝트명: 브레인 매니저

팀명: Thinker

팀원 명단

컴퓨터과학과 2015010925 도우진

컴퓨터과학과 2015010941 장예찬

컴퓨터과학과 2017010831 김정민

경영학과 2017011083 하늘

목 차

소프트웨어 프로젝트 최종보고서

1조 Thinker(Brain Manager)

1. 프로젝트 개요

1.1 프로젝트의 배경 및 필요성

인간의 기억 능력은 일정 주기마다 감소하여, 반복 학습이 매우 중요하다. 이는 에빙하우스의 망각 곡선 이론을 통해 널리 알려진 사실이다. 또, 기억할 때 무작정 외우기보다 기존에 학습된 것과 관련지어 학습하는 편이 더욱 효과적이라는 사실을 이론적, 경험적으로 알고 있다. 따라서 이러한 이론을 반영하여 인간의 기억 능력을 효율적으로 보조하며, 쉽고 강력하게 학습할 수 있도록 어플리케이션을 개발할 필요가 있다.

1.2 프로젝트의 목표와 주요 내용

1.2.1 프로젝트 목표

사용자가 기억하고자 하는 내용(키워드)을 입력받아 망각 곡선에 유의미 학습에 따른 반복 학습을 통해 기억을 보조하는 안드로이드 앱을 제작한다.

1.2.2 프로젝트의 내용 및 범위

- 키워드, 키워드 관련 부연 설명 입력
- 해당 키워드를 망각 곡선 주기에 따라 퀴즈 제시(유의미 학습 이용)
- 사용자의 정답률, 풀이 속도에 대한 통계 그래프 제시
- 사용자의 망각의 정도에 따라 각자에게 일치하는 망각 곡선을 제시
- 학습 형태의 종류 선택

1.3 프로젝트의 업무 분장 및 수행 일정

1.3.1 추진 체계

계획된 역할분장 :

- 도우진(조장) : 설계, 메인화면, DB 구현, 객체 관리, 정렬, 검색 기능 구현 및 전체 보조, 개인정보 처리방침과 오픈소스 라이선스 페이지 작성
- 장예찬 : 화면 설계 초안, 망각 곡선 및 유의미 기억법을 이용한 복습 기능 구현 (복습 간격 알고리즘 문헌 참고 및 개선, 기능 작성)
- 김정민 : 설정, 통계(그래프를 통한 시각화 뷰 지원) 기능 구현

- 하늘 : 발표 PPT 작성, 화면 디자인 보조, 푸시 알림, 잠금 화면, 알람 설정 기능 구현

1.3.2 추진 일정

일정은 기존 목표에서 조금 연기되어 다음과 같이 수행하였다.

- 화면 구성 : 3월 24일까지. 기능 구현은 제외하고 레이아웃 및 뷰 구현과 데이터 객체, 데이터베이스 구현.
- 기능 구현 : 5월 26일까지. 담당한 모든 기능을 구현.
- 베타 출시 및 QA : 5월 26일~

1.3.3 실제 개별 업무 수행

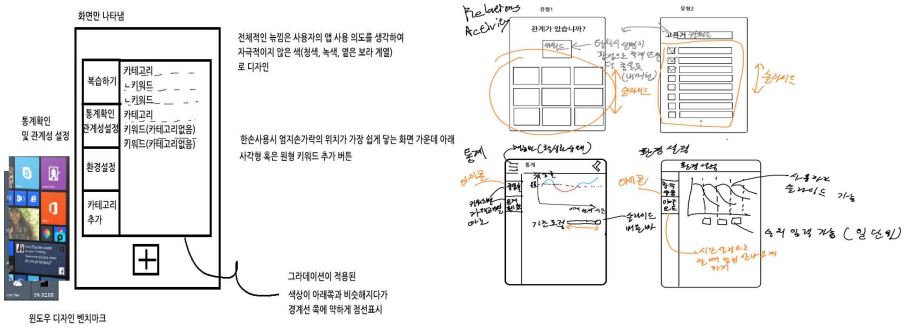
도우진

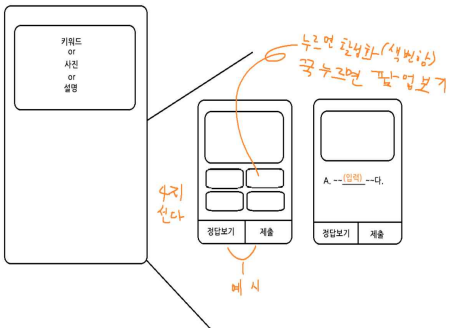
~3/12 : 각 팀원 세부사항 파악, 팀명, 프로젝트 아이디어, 어플리케이션 이름 선 정
3/13 : 주제 발표
~3/23 : 화면설계(어도비 XD), 프로젝트 작업환경 구축(깃 리포지토리 생성 등), Git 관련 안내
~4/1 : 메인화면 기본 UI, Data클래스와 DB작성, 설계서 작성 및 설계 내용 안내
4/2 : 화면 설계(어도비 XD)완성
~4/15 : 메인화면 리스트 동작과 UI작성, 팀 작업내역 통합 및 코드리딩과 리팩터링
~4/28 : 키워드와 카테고리의 추가/제거 구현
~5/8 : 카테고리별 보기 기능 구현, DB 변경, 리스트 표시 시 정렬되게 변경, 키워드 이미지 선택기능 구현, 키워드 검색기능 구현, 중간보고서 작성
~5/12 : 설계서 작성
~5/19 : 카테고리 수정기능 추가, 개인정보 처리방침 및 API 라이선스 페이지 구현, 작업내역 통합 및 Git 관리
~5/26 : 앱 아이콘 제작, 버그픽스, 코드리딩, 리팩터링, 베타 릴리즈
~6/5 : 버그픽스, 코드리딩, 리팩터링, SettingsActivity가 SharedPreference를 활용하여 설정 값 영구저장 및 타 클래스에서 응용 가능하도록 구현, 키워드 이미지 추가 시 원본을 복사하여 복사본 저장하는 기능, 복사본 저장 시 사용자 설정에 따라 원본 이미지 삭제하는 기능 구현, 설정에서 알람과 잠금 화면 선택지가 비활성화 되도록 변경 등, 타 팀 소프트웨어 테스트 및 댓글작성 완료
~6/13 : 시험
~6/16 : SettingsActivity 리디자인, 최종보고서 협동 작성

장예찬

~3/12 : 프로젝트 초안 작성, ppt·제안서 검토

결과물

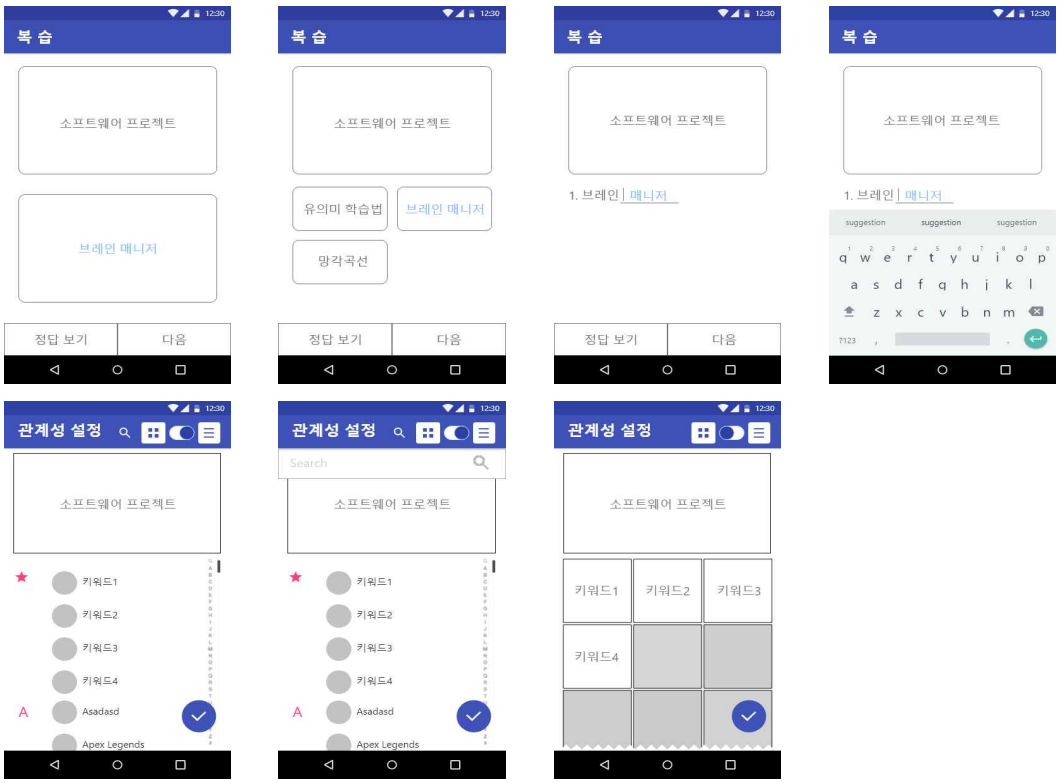




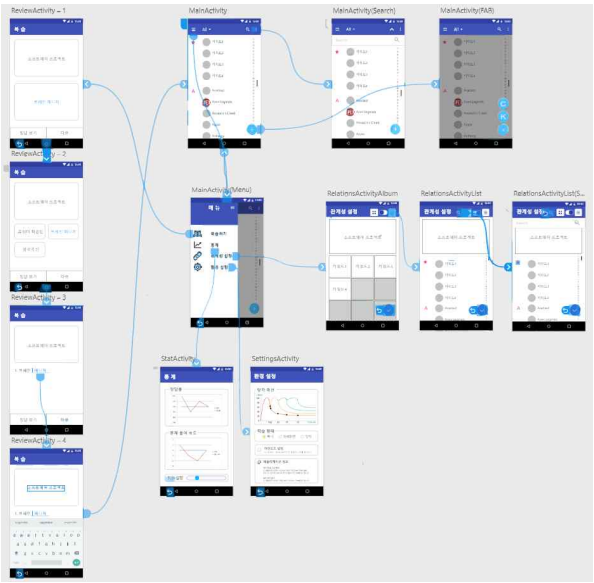
3/13 : 주제 발표

~3/22 : 화면설계(어도비 XD)에 맡은 업무 파트 추가

추가한 결과물



~4/1 : 안드로이드 프로젝트에서 복습하기, 관계성 설정 화면 초기 UI 작성 완료.
ppt·설계서 작성 보조 및 검토.
4/2 : 화면 설계(어도비 XD) 화면흐름도 가시화.
결과물



~4/15 : 안드로이드 프로젝트 복습하기, 관계성 설정 기본동작(터치동작) 구현완료
~4/21 : 시험
~4/28 : 복습·관계성 설정 내부동작(알고리즘 등) 초기 구현완료.
~5/5 : 복습·관계성 설정 시 작은 키워드 설명들 확대 팝업보기 구현완료.
~5/8 : 중간발표 준비 및 ppt·중간보고서 작성 보조 및 검토.
~5/12 : 안드로이드 프로젝트 디자인 변경 및 ppt·설계서 작성 보조 및 검토.
~5/19 : 안드로이드 프로젝트 버그 수정 및 객관식 랜덤출제 알고리즘 수정 및 구현 완성
~5/26 : 각 조원들 기능 병합되자마자
버그픽스, 코드 리팩터링, 베타 릴리즈,
메인메뉴에 키워드 추가할 시 20분 후 복습 알림이 지정되도록 기능 삽입.
프로젝트에서 알림기능 코드(하늘) 재작성 및 리디자인.
ppt 작성 보조.
~6/5 : 안드로이드 프로젝트 알림기능 기능개선(미 복습 시 재 알림 기능 추가)
만료된 복습 존재 시 메인메뉴 내비게이션 아이템 추가하여 간편하게 복습할 수 있도록 구현.
타 팀 소프트웨어 테스트 및 댓글작성 완료
~6/13 : 시험
~6/16 : 최종 댓글 확인 후 요구사항에 따라
튜토리얼(앱 처음 킨 경우) 추가 및 기능 변경, 최종보고서 협동 작성.

김정민

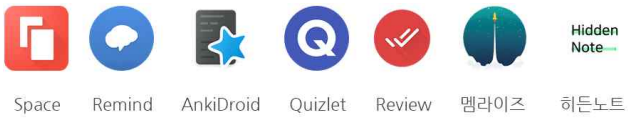
~3/12 : 그래프라이브러리 공부
~4/1 :통계화면 및 설정화면 UI 제작
~4/8 : 설계서 발표
~4/14 : 통계화면 그래프의 스크롤바를 이용한 기간설정 구현
~4/21 : 시험
~5/11 : 스크롤바를 이용하던 기간설정을 스피너형태로 수정 및 완료
~5/20 : 설정화면 UI 수정
~6/13 : 시험

하늘

~3/12 : ppt 제작
~4/1 : ppt 제작
4/13 : 제안서 발표
~4/21 : 작업에 필요한 요소 공부
~5/8 : ppt 제작
~5/25 : 배너 알림 및 알람 구현
~5/27 : ppt 제작 및 동영상 제작
~6/5 : 타 팀 소프트웨어 테스트 및 댓글작성 완료
~6/13 : 시험

2. 기존 현황과 차별성

2.1 국내외 관련 현황



2.2 기존 SW와의 차별성

1. 학습 여부에 따라 자동으로 주기가 조정된다.
2. 다양한 복습 방식 제공(문제 출제: 주관식2형, 객관식)
3. 다양한 알림 기능(현재 푸시알림 구현, 차후 알람, 잠금 화면 형 지원예정)
4. 에빙하우스의 망각 곡선에 유의미 학습을 추가 적용하여 능률 향상.
5. 한글과 영어를 제대로 지원.

	지원 언어		복습간격	알림기능	문제출제기능	키워드 추가방식	온라인 스토리지
	한글	영어					
Space	×	○	단순간격	단순간격 배너만	×	사용자가추가, 글자만.	○
~~Remind~~	최근에 단어 복습 기능이 삭제됨						
AnkiDroid	△번역 잘못됨	○	망각곡선	×	×	사용자가 추가. 글자, 이미지, 음성	○
Quizlet	×	○	×	고정시간 사용자가 배너 알림만 사용자가 지정	×	사용자가 추가 글자, 이미지	×
~~Reivew~~	삭제됨						
멤라이즈	o	×	단순간격	고정시간 사용자가 배너 알림 지정	객관식, 주관식 전체 맞추기	영어단어만 사전에 미리 추가됨. 사용자가 추가 불가능	○
히든노트	○	×	망각곡선	고정시간에 배너 알림만 사용자가 지정	×	사용자가 추가. 이미지만 가능함	○
브레인 매니저	○	○	**망각곡선 + 유의미학습 기반**	**반복간격이 만기됐을 시 배너알림을 생성시킴**	**키워드 이름 맞추기, 객관식, 주관식 설명 일부분 맞추기**	**사용자가 추가, 글자, 이미지**	×

지원하는 항목과 질이 온라인 스토리지를 제외하고 다른 앱과 비교하여 훨씬 우위를 지닌다. 온라인 스토리지와 현재 부족한 통계 기능부분만 보완하면 어떤 타 어플리케이션보다 질적 우위를 가지게 된다.

3. 시스템 구성

3.1 프로젝트 구현 환경

Development O/S	Windows10
Target Platform	Android (API level 23~28)
IDE	Android Studio 3.3.x (Gradle Vesion : 3.3.2)
Language	JAVA (JDK 1.8.x) (람다식 사용)
Version Control	Git & Github

사용된 라이브러리

1.PhotoView

배포 URL	https://github.com/chrisbanes/PhotoView
사용 목적	이미지 줌인/줌아웃 기능 구현
라이선스	Apache License 2.0

2.RecyclerView Animators

배포 URL	https://github.com/wasabeef/recyclerview-animators
사용 목적	RecyclerView에서 아이템 로드 시 동적 애니메이션 효과
라이선스	Apache License 2.0

3.glide

배포 URL	https://github.com/bumptech/glide
사용 목적	이미지 로더
라이선스	https://github.com/bumptech/glide/blob/master/LICENSE

4.ImagePicker

배포 URL	https://github.com/nguyenhoanglam/ImagePicker
사용 목적	저장소에서 이미지를 선택하기 위함
라이선스	Apache License 2.0

5.CircularImageView

배포 URL	https://github.com/lopspower/CircularImageView
사용 목적	둥근 ImageView 구현
라이선스	Apache License 2.0

6.MPAndroidChart

배포 URL	https://github.com/PhilJay/MPAndroidChart
사용 목적	망각 곡선, 정답률, 문제 풀이 속도 차트 표현
라이선스	Apache License 2.0

7.FloatingActionButtonSpeedDial

배포 URL	https://github.com/leinardi/FloatingActionButtonSpeedDial
사용 목적	여닫는 FloatingActionButton 구현
라이선스	Apache License 2.0

8.MaterialEditText

배포 URL	https://github.com/rengwuxian/MaterialEditText
사용 목적	Material 디자인의 EditTextView 구현
라이선스	Apache License 2.0 (저장소의 README.md 참조)

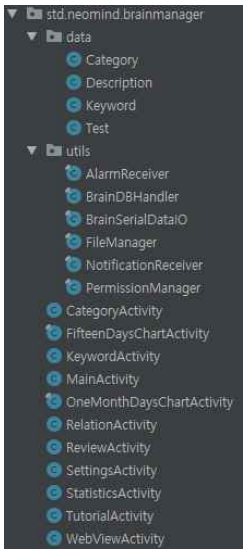
9.SlidingIntroScreen

배포 URL	https://github.com/MatthewDavidBradshaw/SlidingIntroScreen
사용 목적	페이지를 넘기는 방식의 인트로를 통한 튜토리얼 구현
라이선스	Apache License 2.0

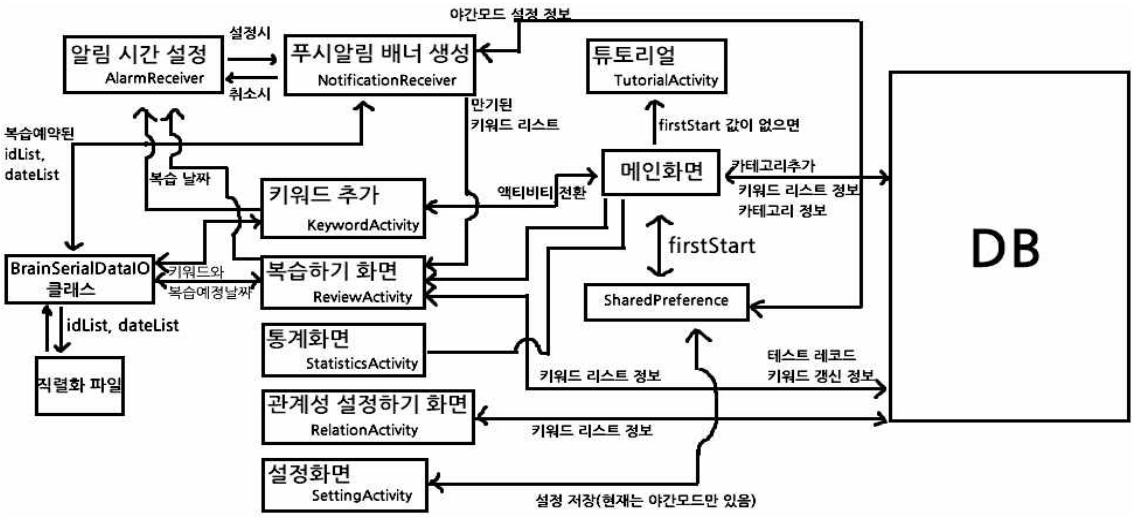
10.AndroidUtilities

배포 URL	https://github.com/MatthewDavidBradshaw/AndroidUtilities
사용 목적	코딩의 수고를 줄이기 위한 여러 Helper 클래스의 사용
라이선스	Apache License 2.0

3.2 기능 구성



Class hierarchy는 왼쪽과 같다.
각 클래스는 MVC(Model, View, Controller) 패턴으로 구분하였다. Model(Category, Description, Keyword, Test)은 data 패키지, View(~Activity)는 기본 패키지, Controller(~er, BrainSerialDataIO)는 Helper 패키지에 포함되었다.



튜토리얼 기능

처음 앱을 켤 때 발생하는 튜토리얼 기능을 기술한다.
최종발표 이후 댓글로 제기된 요구사항을 구현한 기능이며 사용자가 처음 앱을 켤 때 앱의 동작원리를 사용자가 숙지할 수 있도록 도와주는 역할을 한다.

앱이 처음으로 켜질 때 MainActivity에선 SharedPreferences를 통해 "First Start" 값이 false인지 true인지 검사한다. 만약 이 값이 없으면 true를 반환하며
true이면 TutorialActivity를 실행하고 아니면 MainAcitivity를 그대로 실행한다.

이를 좀 더 세련되고 깔끔하게 제공하기 위해 오픈소스인
matthew-tamlin:sliding-intro-screen 라이브러리와
matthew-tamlin:android-utilities 라이브러리를 사용한다.

android-intro-screen은 IntroActivity라는 Activity컴포넌트를 상속하는 클래스를 가지고 있는데 이를 통해 제공되는 메서드와 android-intro-screen에서 추가적으로 제공하는 클래스들을 통해 깔끔하고 좋은 UX를 가진 튜토리얼을 사용자에게 제공한다.

이 라이브러리에선 Fragment를 상속한 ParallaxPage 클래스를 제공하며 이는 튜토리얼에서 각 페이지를 뜻하는 것이다.
이를 다음과 같이 사용할 수 있다.

```
ParallaxPage newPage = ParallaxPage.newInstance();
newPage.setFrontImage(frontBitmap);
newPage.setBackImage(backBitmap);
```

여기서 frontBitmap과 backBitmap은 Bitmap클래스의 객체이며 png의 투명도를 활용해서 앞과 뒤의 Bitmap을 튜토리얼을 넘기는 애니메이션이 발생할 때 동적으로 위치가 변하면서 더 보기 좋은 애니메이션으로 만들어준다.

IntroActivity클래스는
Collection<? extends Fragment> generatePages(Bundle savedInstanceState) 라는 인터페이스를 반드시 오버라이드 구현해야하며 위와 같이 ParallaxPage 객체를 작성하고 어레이 리스트에 삽입하여 튜토리얼에서 보여줄 페이지를 늘릴 수 있으며 마지막으로 ArrayList<Fragment>를 반환하는 식으로 구현해야 한다.

```
final ArrayList<Fragment> pages = new ArrayList<>();
pages.add(newPage);
```

위와 같이 페이지를 추가할 수 있고 마지막 return pages; 를 통해 이 함수를 종료한다.

이 사용법에 맞게 이 메서드의 구현에서 android-intro-screen에서 제공하는 클래스인 MultiViewParallaxTransformer와 BackgroundManager 클래스를 통해 화면에 Blending효과(부드럽게 백그라운드 컬러를 변경)를 주었고
각 컬러들은 Adobe Color에서 Best Color 조합 순위에 들어가는 것을 사용하여 지정했다.(0xff33B7A9, 0xff026D67, 0xff025E73, 0xff468535)
그 이후 png파일들을 프로젝트의 리소스로 추가하여 두개 의 Bitmap Array(크기4)로 만들었다.
matthew-tamlin:android-utilities을 사용하여 ScreenSizeHelper클래스의 메서드를 통해 현재 기기의 가로 세로 화면의 크기를 받아오게 저장하였고
매개변수로 전달된 가로 세로 크기에 맞게 비트맵의 사이즈를 지정해주는 BitmapEfficiencyHelper.decodeResource(context, resource id, 가로크기, 세로크기) =>반환값 Bitmap메서드를 통해 BitmapArray들을 초기화 했다.
frontBitmapArray의 각 원소가 가지고 있는 파일
R.raw.front_start_tuto,
R.raw.front_main_tuto,
R.raw.front_review_tuto,
R.raw.front_relation_tuto
(전부 png파일)
backBitmapArray의 각 원소가 가지고 있는 파일
R.raw.back_start_tuto,
R.raw.back_main_tuto,
R.raw.back_review_tuto,
R.raw.back_relation_tuto
(확장자는 모두 png)
이들을
```
ParallaxPage newPage = ParallaxPage.newInstance();
newPage.setFrontImage(frontBitmapArray[i]);
newPage.setBackImage(backBitmapArray[i]);
pages.add(newPage);
```
와 같은 방식으로 반복문을 통하여 pages Fragment 리스트를 추가하고 이를 반환했다.

또한 IntroActivity클래스는 반드시 IntroButton.Behaviour
generateFinalButtonBehaviour()을 오버라이드 구현하여 튜토리얼 마지막에서
완료버튼의 동작을 작성하고 IntroButton.Behaviour르 반환해야한다.

완료버튼의 동작으로는 SharedPreferences 클래스를 사용하여 SharedPreferences 저장소에 "First Start" 라는 값을 false로 만드는 식으로 구현되어 있다.
반환 값으로는
IntroButton이라는 클래스의 이너 스태틱 클래스(IntroButton.BehaviourAdapter를 상속한)로 기본 동작들을 가진 클래스들을 제공하는데 이 라이브러리 소스를 보았을 때 finish()를 구현한 기본 동작이 존재하지 않기에
IntroButton.BehaviourAdapter를 상속한 static 클래스인 DoFinish를 직접 작성하여 이 클래스를 사용하여 반환 값으로 사용하였다.
(IntroButton.BehaviourAdapter는 IntroButton.Behaviour를 상속하고 있다.)

메인화면의 기능

키워드 추가 기능

키워드 추가 기능의 설명은 MainActivity의 오른쪽 하단 FloatingActionButton을 터치해서 키워드 추가를 터치했을 때에 생성되는 KeywordActivity 화면 컴포넌트 객체의 동작을 설명한다.

DB와 레코드(Model) 클래스

1. Keyword 클래스

위 클래스는 DB에 키워드를 저장하거나 불러올 때, DB의 레코드들을 담는 클래스이다.
이 클래스는 다음과 같은 필드들을 가지고 있다.

private CardView cardView ← 키워드객체가 표시되는 레이아웃을 저장하는 용도
public int id; ← 키워드 id(주키)
public int cid; ← 키워드 카테고리 id(외래키)
public String name; ← 키워드 이름
@NonNull private ArrayList<Description> descriptions; ← 키워드 설명
public String imagePath; ← DB 또는 저장소에 저장된 이미지 링크
public int currentLevels; ← 현재 주기 레벨(알고리즘에 사용)
public int reviewTimes; ← 복습 횟수(알고리즘에 사용)
public long registrationDate; ← 추가된 날짜(통계에 사용)
@NonNull private ArrayList<Integer> relationIds; ← 관계성List (알고리즘에 사용, 자기 참조 외래키 집합)
public double ef; ← 기억용이성(알고리즘에 사용)
public int interval; ← 복습 간격
private boolean selected; ← 키워드의 선택(메인화면, 관계성화면)을 위한 임시 필드

이 중 DB에 저장되는 필드들은 id, cid, name, imagePath, currentLevels, reviewTimes, registrationDate, ef, interval이다.

descriptions와 relationIds는 DB 테이블에 존재하는 설명(String)리스트와 관계성 있는 키워드들을 불러오는 id값들이다.
나머지 cardView와 selected는 DB에 저장되지 않고 키워드 객체를 생성할 때 초기화 되는 필드이다.

2. Description 클래스

설명 Stirng 리스트를 위해 DB에 저장되는 레코드 클래스이다.
다음과 같은 필드를 가지고 있다

```
public int id;                  ← 키워드 id(주키)
public String description;      ← 설명
```

키워드 id를 주키로 사용하며 BrainDBHandler를 통해 DB의 Description테이블에서 데이터를 불러와 이 객체를 생성한다. 이 객체는 Keyword객체를 DB에서 불러올 때 DB에서 불러와져 ArrayList<Desciption> 형태로 Keyword객체에 저장된다.

3. Test 클래스

사용자가 복습하기(ReviewActivity)를 통해 문제를 풀었을 때 이 문제의 통계를 내기 위해서 DB에 저장하여 사용하는 레코드 클래스이다.

```
public int id;              ← 테스트 레코드의 id(주키)
public int cid;             ← 카테고리 id(외래키)
public int kid;             ← 키워드 id(외래키)
public long testedDate;     ← 문제를 푼 날짜(=이 객체가 생성된 날짜)
private boolean passed;     ← 문제 합격여부
public int answerTime;      ← 문제 풀이속도
public int type;            ← 문제 유형
```

하지만 이 레코드는 통계기능의 세부적인 구현이 미흡하여 현재는 기록만하고, 실제로 사용되지는 않고 있다.

4. BrainDBHandler

레코드 클래스들을 DB에 저장될 테이블 형태로 제구성하고 저장시키는 클래스이다. DB에 레코드 클래스를 저장하거나 DB에서 데이터 레코드를 검색하여 레코드 클래스의 형태로 반환하는 등, DB에 질의 명령을 단순하게 사용할 수 있도록 다양한 메소드가 제공된다. 이는 안드로이드 내장 클래스인 SQLite의 기능 중 하나인 SQLiteOpenHelper를 상속하는 클래스이다. Exception 클래스를 상속받는 클래스들(NoMatchingDataException, DataDuplicationException)을 inner class로 두어 찾는 데이터가 없거나, 키워드의 관계성 집합을 별도의 테이블로 나타낸 Relations 테이블에서 클래스단계에서의 무결성 유지에 위배가 발생할 경우 예외를 throw하여 해

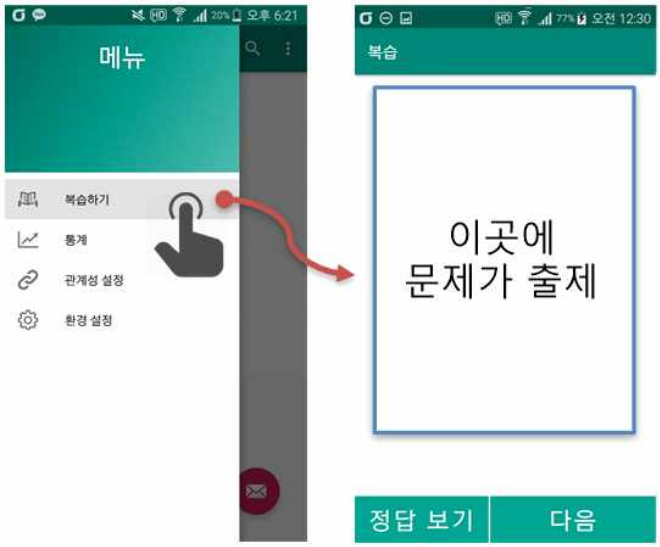
당 메소드를 사용하는 부분에서 적절히 대응할 수 있도록 구성하였다.

복습하기 기능

복습하기 기능의 설명은 복습 간격 알고리즘을 제외하고 메인메뉴 내비게이션에서 복습하기를 눌렸을 때의 MainActivity 화면 컴포넌트 객체와 ReviewActivity 화면 컴포넌트 객체의 동작을 기술한다.

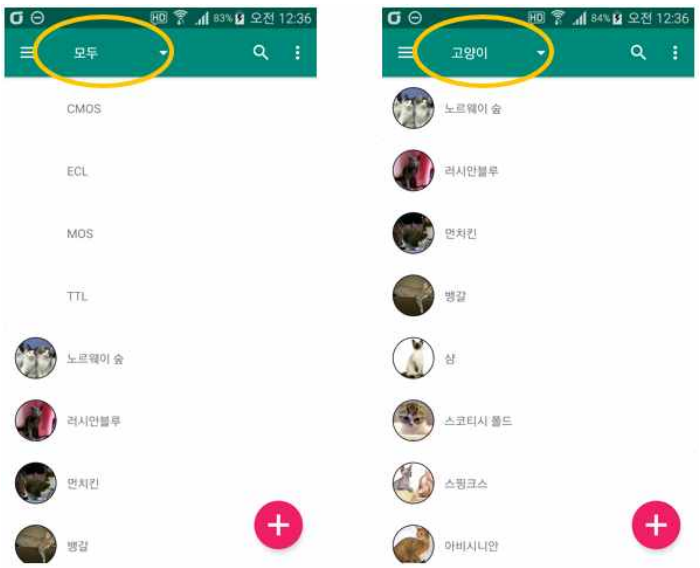
두 객체는 안드로이드의 Activity를 상속하는 클래스의 객체이다.

1. MainActivity의 내비게이션에서 '복습하기' 터치했을 때의 동작



("이곳에 문제가 출제"는 임의로 넣은 텍스트 상자임)

이 이벤트가 발생하면 MainActivity에서 현재 보고 있는 카테고리의 키워드들을 Intent에 id를 기반으로 ArrayList<Integer>에 저장하고 직렬 화하여 전달한다. (Keyword자체는 DB와 클래스 설계 당시 CardView 필드로 인하여 직렬 화가 이루어지지 않기 때문에 이 방식을 사용한다.)

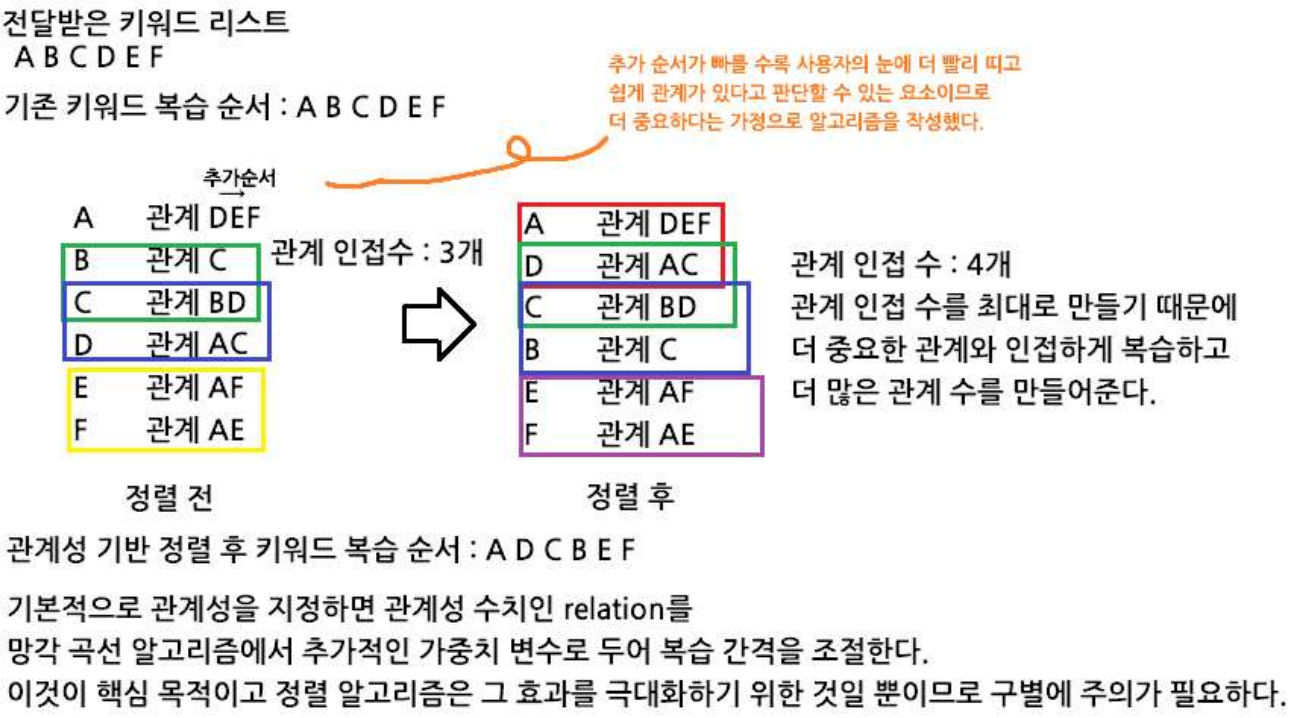


(MainActivity)
현재 보고 있는 카테고리
(노란색 동그리미)의
키워드id들을
ReviewActivity로 전달한다.

ReviewActivity에서는 전달된 키워드 id ArrayList를 사용하여 DB핸들러 메서드를 통해 **DB에서 해당 키워드 리스트를 불러온다.**
만약 이 키워드 리스트가 0개라면 ReviewActivity가 종료되며 MainActivity로 돌아가 키워드가 없다는 메시지를 화면에 Toast로 출력한다.

키워드가 없을 시
잠시 출력되는 메시지

ReviewActivity는 DB에서 불러온 해당 키워드 리스트를 관계성 기반으로 재 정렬한다. 서로 관계있는 키워드가 최대한 인접하게 출제되도록 최대 관계 정렬 알고리즘을 직접 작성하여 정렬하였다. 그 다음, 이 키워드 리스트(이하, 문제 키워드 리스트)를 순서대로 문제로 출제한다.



<최대 관계 정렬 알고리즘의 동작 예시>

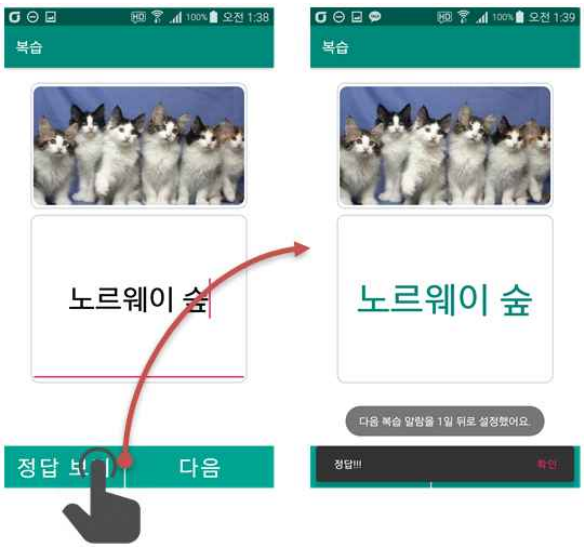
이는 유의미 학습의 참고한 논문 주요 내용상 관련된 것을(개념적 하위내용, 같이 배운 내용 등등) 동일시기에 학습하거나 복습하면 뇌의 시냅스 그물 구조상 더 높은 학습 성취를 얻을 수 있다는 점에 영감을 얻어 작성된 정렬 알고리즘이다.

또한 DB에서 **전체 키워드 리스트**를 불러오고 (키워드 객체는 int와 string 몇 개만을 필드로 가지므로 굉장히 가볍다) 이 리스트에서 문제로 출제 가능한 키워드(=설명 이미지나 설명 글이 존재하는 키워드,)의 개수(이하, 설명 존재 개수)를 센다.

설명이 존재하지 않는다면 문제로 출제가 불가능하다는 뜻이므로 위와 같이 ReviewActivity가 종료되며 MainActivity로 돌아가 키워드가 없다는 메시지를 화면에 Toast로 출력한다.

이 설명 존재 개수와 키워드의 설명유형을 통해 출제 유형이 변할 수 있다.
기본 문제 출제 유형은
(1) 키워드 이름 맞추기 (2) 올바른 설명 고르기 (객관식)
(3) 키워드 글 설명에서 빈칸 채우기
의 세 가지 방식이다. 이 세 가지 방식에서 랜덤으로 문제를 출제한다.
출제 방식에는 조건이 있다.
설명 존재 개수가 2개 미만일 때는 2. 객관식 출제가 불가능하고, 키워드의 설명 유형이 이미지 설명만이 존재하면 3. 글 설명에 빈칸 채우기는 출제가 불가능하다.
키워드에 설명 자체가 존재하지 않는다면 문제 출제 자체가 불가능하다.
문제 출제가 불가능할 경우에는 키워드 리스트의 다음 키워드로 문제를 출제한다.
각 출제 유형별로 조건을 불만족하면 해당 유형은 출제 유형에서 제외된다.

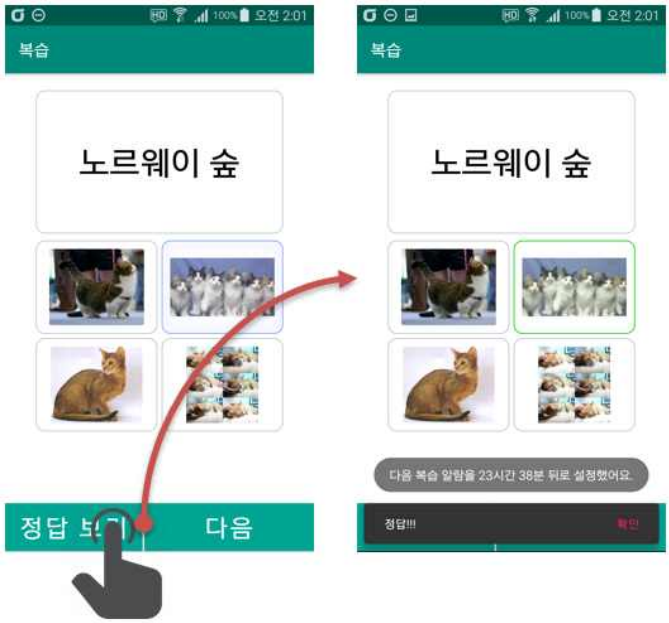
(1) 키워드 이름 맞추기



노르웨이 숲을 입력하고
정답 보기를 누르자 정답이라고
표시되고 다음 알림이 지정됨.

키워드의 설명(위)을 보고 빈칸(아래)인 키워드의 이름 EditText을 작성하여 정답을 맞히는 문제이다.
설명은 두 유형(글, 이미지) 모두 출제될 수 있다. 두 유형의 설명이 모두 존재하면 랜덤으로 하나만 표시한다.
위 예시에선 고양이 종(노르웨이 숲)을 외우기 위해 복습하고 있다.

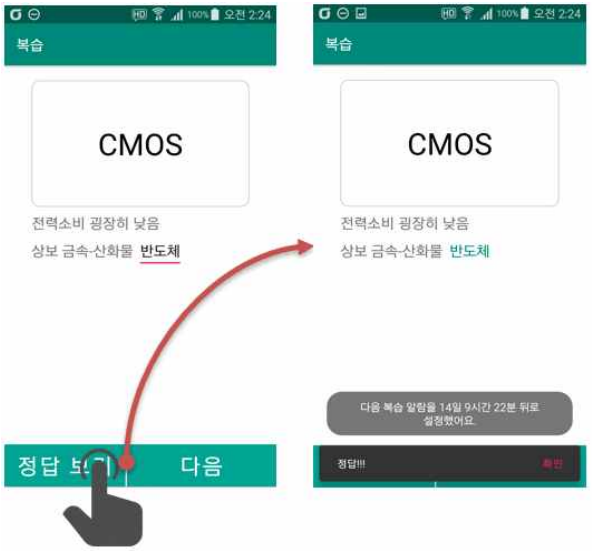
(2) 올바른 설명 고르기 (객관식)



오른쪽 위가 선택되어 있고 정답보기를 누르자 정답이라고 표시되고 다음 알림이 지정됨.

키워드 이름(위)을 보고 해당하는 올바른 키워드의 설명을 선택해서 정답을 맞히는 문제이다.
설명은 두 유형(글, 이미지) 모두 출제될 수 있다. 보기는 일단 키워드의 필드인 관계성 id 리스트를 DB에서 키워드리스트를 불러와서 관계성이 지정된 키워드들을 우선하여 랜덤한 위치에 지정되며 이후엔 카테고리를 우선하여 랜덤한 위치(4지선다)에 지정된다.
관계성이 있는 키워드가 보기에 랜덤으로 나오는 이유는 유의미 학습의 관련 논문 주요 내용인 관련된 내용들을 학습할 때 동일시기에 학습하거나 복습해야 이들의 **다른 점을 구별할 수 있는 능력**이 강화된다는 것에서 영감을 얻어 작성한 메커니즘.
위치의 개수(2~4)는 설명 존재 개수와 같으며 4개 이상 존재할 경우 위치의 개수는 4로 고정된다. 위치의 개수를 통해 설명 존재 개수가 4개 미만일 때 빈칸의 위치가 바뀌는 것을 방지한다.
위 예시에선 고양이 종(노르웨이 숲)을 외우기 위해 복습하고 있다.

(3) 키워드 글 설명에서 빈칸 채우기



주어진 빈칸에 반도체라고 입력하고
정답보기를 누르자 정답으로
표시되고 다음 알림이 지정됨.

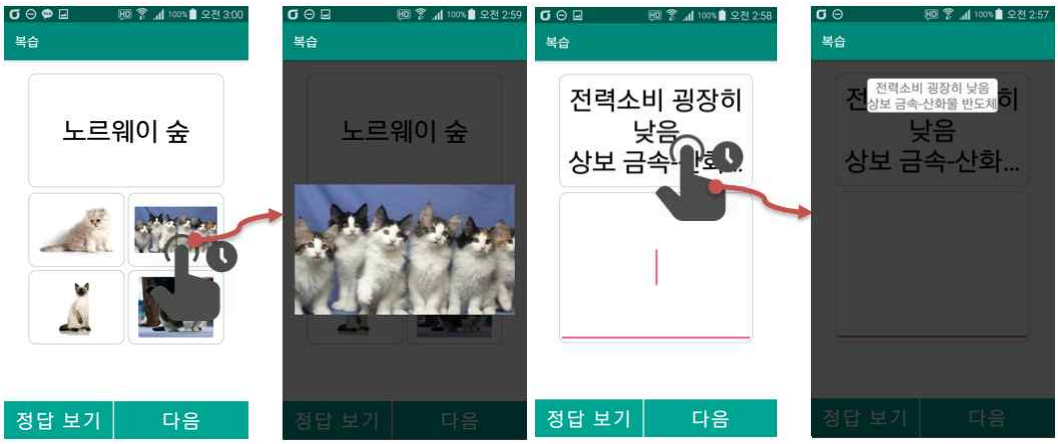
키워드 이름(위)을 보고 주어진 설명(아래)에서 빈칸을 올바르게 채워 넣는 문제이다.
설명은 글 설명만 출제될 수 있다.
각각의 설명 줄은 동적으로 하나의 LinearLayout으로 생성하며 이 레이아웃에 각 ArrayList<String>의 차례대로 한 아이템을 TextBox로 삽입한다.
특별히 빈칸이 존재하는 줄은 LinearLayout에 3가지 경우의 조건 구조를 가진다.
①TextBox(빈칸 앞내용) + EditBox(빈칸) : 빈칸이 끝에 존재할 경우
②EditBox(빈칸) + TextBox(빈칸 뒷내용) : 빈칸이 맨 앞에 존재할 경우
③TextBox(빈칸 앞내용) + EditBox(빈칸) + TextBox(빈칸 뒷내용) : 빈칸이 중간에 존재할 경우
위 예시에선 컴퓨터 구조의 CMOS를 외우기 위해 복습하고 있다.

2. ReviewActivity에서 '정답 보기'와 '다음'의 눌렸을 때의 동작

문제는 정답 보기 버튼을 누르기 전까지는 다음 버튼을 누를 수 없게 되어 있다.
정답 보기 버튼을 누르면 현재 문제가 정답인지 아닌지를 판단하고 정답과 오답을 화면에 표시하며
정답여부를 Snackbar형태(위 예시 이미지에서 아래쪽 정답!!! 확인 Bar) 로 출력한다.
복습 간격 알고리즘을 사용한 다음 복습시간 반환(long타입 기간) 메서드를 호출하여 다음 복습시간을 AlarmReceiver라는 브로드캐스트 리시버로 전달하여 알림을 설정한다. 그리고 BrainSerialDataIO라는 스태틱 메서드들을 가진 클래스를 사용하여 앱의 내부 저장소에 현재 키워드의 id와 다음 복습 시간을 각각 ArrayList<Integer>와 ArrayList<Long>타입으로 직렬 화해서 저장한다. 이 클래스에 대한 것은 아래쪽 '객체 직렬 화 저장 기능'에서 기술한다) 이 저장된 객체들을 이후에 키워드의 복습 만료 여부를 알아낼 때 사용될 것이다. 또한 이 클래스의 스태틱 메서드를 통해 얻

은 현재 키워드의 복습 예정 시간이 만기상태이면 해당하는 id integer와 long 아이템을 직렬 화 파일에서 제거한다.
이후에 다음 버튼을 누르면 문제 키워드 리스트의 다음 키워드를 위의 기술한 방식대로 문제로 출제한다.
문제 키워드 리스트의 맨 마지막 키워드에서 정답을 확인 후 다음 버튼을 누르게 되면 ReviewActivity가 종료되며 MainActivity로 돌아가 맨 마지막 문제라는 메시지가 화면에 Toast로 출력된다.

3. ReviewActivity에서 설명을 오래클릭=롱클릭(LongClick) 했을 때의 동작.



ReviewActivity에서 설명들은 꾹 누르는 동작 이벤트(LongClick 이벤트)가 발생했을 때, 어떤 유형의 설명이든 전체를 볼 수 있다. 이는 사용자의 편의성과 불편함을 줄여 쾌적한 UX를 제공하기 위해 구현된 기능이다.
여기서 이미지를 전체보다 더 확대해서 보기 위해 chrisbanes:PhotoView 오픈소스 라이브러리가 사용된다. 이는 ImageView를 상속하여 구현된 클래스이며 이미지를 쉽게 확대하고 축소할 수 있게 최적화된 기능을 가지고 있다.

4. ReviewActivity에서 뒤로 가기(BackPress) 버튼을 눌렀을 때의 동작.

ReviewActivity에서 뒤로 가기를 눌렸을 경우, 만약 3.의 설명 전체보기가 활성화되어 있을 경우엔 이를 닫는 동작을 수행하고 아닐 경우엔 위와 같이 두 가지 경우의 종료할 것인지를 묻는 AlertDialog가 생성된다. 좌측은 키워드 중 만기된 복습이 하나도 없는 경우에 생성되는 AlertDialog이고 우측은 만기된 복습이 하나라도 있는 경우 생성되는 AlertDialog이다.
우측의 경우에 '네' 버튼을 누를 경우 사용자가 제대로 복습을 남긴 것으로 간주하고 AlarmReceiver 브로드캐스트리시버를 취소모드로 작동시키고 이는 20분 뒤에 NotificationReceiver 브로드캐스트리시버를 발생시킨다. 이 순간에 만약 만기된 복습이 하나도 없을 경우 배너(Notification)가 생성되지 않는다.
즉, 사용자가 취소시켜서 알림이 지정되었지만 그 전에 만기된 복습을 전부 완료하면 NotificationReceiver 브로드캐스트리시버가 제 때 동작하더라도 아무 동작을 하지 않는다는 것이다.
알림 기능의 구조는 이후 알림 기능 부분에서 자세히 기술할 것이다.

복습 간격 알고리즘

이 앱의 핵심 기능인 복습 간격 조절에 대한 동작 원리를 기술한다.

10초 안에 정답일 경우 rating=5
30초 안에 정답일 경우 rating=4
그 외의 정답은 rating=3
5초 이상 10초 이내에 오답일 경우 rating=2
그 외의 오답 rating=0

rating<3일 경우 keyword의 curretLevels를 0으로 초기화
ef는 기억 용이성(기억하기 쉬운 정도)
ef의 초기 값=2.5

매 학습시마다
ef = ef+(0.1-(5-rating)*(0.08+(5-rating)*0.02))
ef가 1.3보다 낮아지면 1.3으로 다시초기화
2.5보다 작아지면 2.5로 다시 초기화

relation은 1.07^(현재문제의 관계있는 보기 수(객관식일 경우) + 이전까지 풀었던 인접한 관계있는 문제 개수)
다음 복습간격 I(currentLevels)는
만약 복습이 만료되지 않았다면 이전 알림시간과 일치하게 간격을 지정. 그 이외는
I(0): 20분*relation

I(1): 24시간*relation
I(2): 6일*relation
I(L): I(L-1) * ef * relation

복습의 만료 여부는 위에서 BrainSerialDataIO로 생성한 로컬의 직렬 화된 파일을 불러와서 복습 예정 시간인 ArrayList<Long>타입을 체크하여 문제를 시작한 시간이 이를 넘었는지를 판별한다.

-요약-

사용자가 정답을 확인하면

정답 등급(rating)에 따라 **기억용이성**(기억하기 쉬운 정도에 비례)가 변하며 또한 **관계성 값**(인접한 관계된 문제, 혹은 객관식 보기 수에 지수비례)을 지정.

만약 복습이 만료되지 않았다면 이전 알림시간과 일치하게 간격을 계산. 그 이외는 오답일 경우 첫 복습 간격부터 시작.

첫 복습 간격은 20분 x 관계성 값

두 번째 복습 간격은 24시간 x 관계성 값

세 번째 복습 간격은 6일 x 관계성 값

네 번째 복습 간격부터는 이전 복습 간격에서 기억용이성과 관계성 값을 곱하여 다음 간격을 계산한다.

복습 간격이 정해지면 AlarmRecevier 브로드캐스트리시버를 호출하여 이를 넘겨주어 다음 복습 시간에 알림이 오도록 등록.

간격 반복 알고리즘 (SM-2 알고리즘 개선)

정답 보기를 누르면 등급에 따라 기억용이성 증감(SM-2의 원칙에 따라)

기존 알고리즘에서의 개선사항으로는 **등급을 축소**하였고(**5가지->4가지**)
사용자가 오답을 내더라도 기억용이성은 **초기화되지 않으며**(기존: 2.5로 초기화됨)
각 복습 간격에 관계성 값을 곱하는 것으로 **유의미 학습을 적용**하여 키워드별 관계성이 추가되었을 때 더 최적의 간격을 계산할 수 있도록 개선함.
이는 유의미 학습의 참고한 논문의 주요 내용상 관련된 것을(개념적 하위내용, 같이 배운 내용 등등) 동일시기에 학습하거나 복습하면 뇌의 시냅스 그물 구조상 더 높은 학습 성취를 얻을 수 있다는 점을 활용해 가중치 변수를 추가한 것이다.

알림 기능

알림기능을 담당하는 AlarmReceiver와 NotificationReceiver에 대해 기술한다.
알림기능은 처음 작성이 완료된 상태(5월25일)에선 요구사항과는 다르게 원하는 시간을 지정할 수 있는 기능(메소드)를 가지고 있지 않았다. 또한 터치 시, 취소시의 동작이 구현되지 않았다.
이는 ring과 main22, BroadcastD2, Broadcast22, SettingsActivity에 걸쳐 작성되어 있었는데 애매한 명명방식으로 의미가 난해했고 미리 지정된 시간에 대해서만 동작하게 작성되어 있었다. 또한 알람 알림의 알람 화면이 제대로 구현되어있지 않았다. 그리고 벨소리 ring클래스가 서비스로 구현되어 있는데 목적대로라면 알람 화면이 표시되고 있을 때만 벨소리가 울리게 할 예정이기 때문에 이 또한 스레드로 구현해야 했었다.

하늘 조원은 경제학과 복수전공의 3학년 상태라서 수강한 컴퓨터과학 강의들의 수가 적기 때문에 코딩 실력의 부족으로 이와 같은 문제가 발생하였다. 따라서 장예찬 조원이 코드들을 분석하고 전반적인 재작성을 했다.
코드 분석 후 시간이 촉박한 관계로 알림 기능 중 알람 방식은 사용하지 않기로 결정했다.
알림기능은 새로 작성한 AlarmReceiver 클래스와 Broadcast22를 재작성한 NotificationReceiver 클래스로 배너알림이 구현되었다. 향후에 ring을 스레드 방식으로 변경하고 알람화면을 재설계하여 알람기능 또한 제공할 것이다.

각각의 기능을 기술하자면 다음과 같다.

1.AlarmReceiver

이 클래스는 BroadcastReceiver 를 상속하고 있으며 이를 상속하고 있으면 onReceive라는 추상 메서드를 구현하여 기기에 특정 이벤트(재부팅, 시간이 바뀔 때)가 발생할 때나 임의로 호출해서 이 메서드를 실행시킬 수 있게 된다.

이 클래스가 구현된 이유는 복습하기에서 정답을 눌렀을 경우, 어떠한 경우에도 정상적으로 알림이 등록되도록 하기 위함이다. 기존에는 알림 시간을 지정할 수

있는 기능이 없었기 때문에 알람을 등록할 코드 내에서 직접 알람을 설정해야한다.

브로드캐스트리시버는 새로운 스레드 형태로 실행되므로 사용자가 정답을 확인했을 경우 어떠한 경우에도 알람이 제대로 등록된다. 이것은 삼성 갤럭시 스마트폰의 알람 앱(원리, 공개된 코드는 없음)에서 구현된 방식과도 같다.

이 클래스는 AlarmManager라는 안드로이드에서 제공하는 클래스 객체를 사용하고 있으며 이는 PendingIntent가 Broadcast 컴포넌트를 받은 객체(컴포넌트의 정보를 가짐)를 받아 특정시간에 해당 컴포넌트 클래스 객체를 동작하게 만든다.

getBroadcast메서드를 통해 PendingIntent를 생성할 때, 매개변수인 requestCode 값이 같을 경우엔 마지막에 실행된 PendingIntent만 동작하므로 이 requestCode값을 (int)(알림 지정 시간/1000/60/10) 와 같이 지정하여 10분 간격으로 requestCode 값이 바뀌도록 메커니즘을 구성하였다.

동작은 3가지 조건이 존재한다.
① 알람시간을 넘겨받고 직접 호출되었을 때
사용자가 ReviewActivity에서 정답을 확인했을 때 발생하며 알림시간을 전달받고 이 리시버가 동작한다. **이 경우 AlarmManager객체를 통해 전달받은 알람시간에 맞춰 NotificationReceiver를 직접 호출한다.**

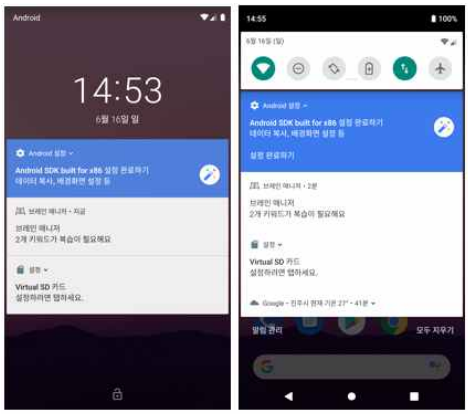
② 취소모드로 직접 호출되었을 때
사용자가 NotificationReceiver에서 생성한 배너알림을 선택하지 않고 임의로 삭제시키는 경우나 복습화면에서 만기된 복습이 있음에도 복습을 완료하지 않고 앱을 중지시킬 경우에 발생한다. 알람시간은 넘겨받지 않는다. **이 경우 AlarmManager객체를 통해 현재로부터 20분 뒤에 NotificationReceiver를 직접 호출한다.**

③ 재부팅 이벤트가 발생했을 때
사용자가 스마트폰을 재부팅했을 때 발생하고 이 리시버가 동작한다. 스마트폰을 재부팅하는 경우 AlarmManager로 지정된 알람들이 모두 제거되므로 이를 다시 재등록해줘야 되기 때문이다. 이 때, **BrainSerialDataIO의 스태틱 메서드를 통해 예정된 알람시간을 불러오며 이 시간들에 맞춰 AlarmManager객체를 통해 NotificationReceiver를 직접 호출한다.**

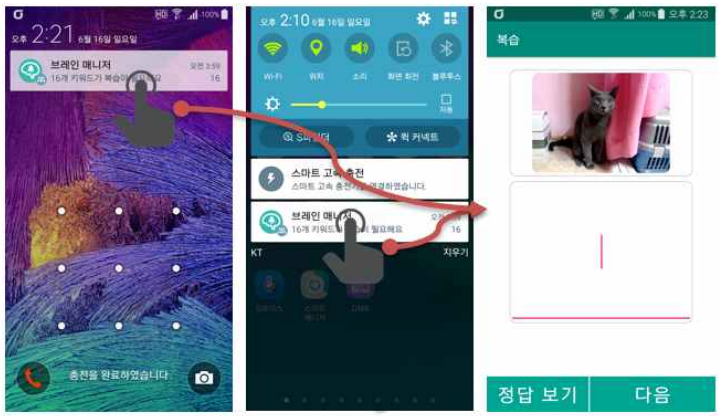
2. NotificationReceiver
이는 Broadcast22를 재작성한 클래스이다.
BroadcastReceiver 를 상속하고 있으며 onReceive를 구현하고 있다. 이 리시버는

직접호출로만 동작하며 기기의 이벤트를 받지는 않는다.
이 리시버가 동작하면 BrainSerialDataIO을 통해 예정된 알람시간을 불러오며 현재시간과 복습 예정시간을 비교하여 만기된 복습의 개수를 센다. **만약 0개라면 배너알림은 발생하지 않는다.** 이는 재작성시 추가되었다.
재작성 후 변경·추가된 기능들을 요약하자면 위를 포함하고, 알림 아이콘 변경, getString 메서드를 통한 기기 언어 영·한 버전 지원, 안드로이드 버전별 동작, 야간 모드 기능, 배너를 터치했을 때 동작, 배너를 취소했을 때 동작이 있다.

이는 NotificationManager라는 안드로이드에서 제공하는 객체를 사용하고 있으며 이를 통해 안드로이드 화면의 상태 바에 배너알림을 띄울 수 있게 한다. NotificationManager 클래스는 안드로이드 버전 오레오 이상과 오레오 미만이 서로 동작이 다르게 되기 때문에 이를 if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) 구문을 통해 버전별 동작을 구현했다. 이와 같이 동작을 다르게 구현하지 않으면 앱의 사용 시 사용자는 예기치 않은 오류를 겪을 수 있다.



Android Oreo 버전 이상



Android Oreo 버전 미만

위와 같이 Oreo버전 이상과 미만은 배너 알림의 생김새가 약간 다름을 알 수 있으며 이는 버전별로 정의 가능한 동작이 다르기 때문에 그렇다. 두 경우 모두 배너알림을 터치했을 때 복습하기 화면으로 이동하는 것은 동일하다.

야간모드 기능은 SharedPreferences 클래스를 사용하여 앱의 SharedPreferences저장소를 불러와서 사용한다. SettingsActivity에서 야간모드를 할 것인지를 묻는 체크박스의 상태 값을 불러와서 체크가 되어 있으면 소리와 진동이 발생하지 않도록 구현했다. 이 또한 안드로이드 오레오 이상과 미만이 서로 다르게 동작하도록 작성되어 있다.

① 배너알림 터치 시 동작
배너알림을 터치했을 경우 ReviewActivity 객체가 생성되며 Activity컴포넌트가 실행

되게 된다. 이 경우 위에서 기술한 **1. MainActivity의 내비게이션에서 '복습하기' 터치** 에서 MainActivity 화면에서 보고 있는 카테고리의 키워드들을 넘기는 것이 아니라 BrainSerialDataIO로 저장된 직렬 화 파일에서 id리스트인 ArrayList<Integer>를 불러와서 이 중 만기된 복습만을 남기는 식으로 동작한다. 나머지 동작은 위에서 기술된 복습하기 기능과 동일하다.

② 배너알림 취소 시 동작
배너알림을 취소했을 경우 AlarmReceiver 브로드캐스트 리시버를 취소모드로 호출한다. 이 때, AlarmReceiver는 20분 뒤에 다시 NotificationReceiver를 실행하는 식으로 동작한다.

객체 시리얼(직렬 화) 저장 기능

BrainSerialDataIO클래스에 대한 것을 기술한다.
이 클래스는 직렬 화 된 데이터를 IO 하는 메서드들을 정의하며 현재 세 가지 static 메서드를 제공한다.
세 메서드들은 복습 키워드 id integer와 복습 예정 날짜를 Long 리스트 형태로 각각 직렬 화해서 내부저장소에 저장하며 이를 통해 주로 현재 어플리케이션에서 복습 예정인 키워드가 일자가 만기되었는지 확인한다.

각 메서드들은 Context를 매개변수로 받는데 이는 내부 저장소에 접근하기 위한 매개변수이다. 저장되는 직렬 화된 파일의 이름은 BrainAlarm.data이다.

1. saveNextReviewTimeInfo 메서드

매개변수로 Context context, ArrayList<Integer> idList, ArrayList<Long> dateList를 받으며 idList는 저장될 키워드들 id 리스트 , dateList는 저장될 복습 예정 날짜 리스트들을 받는다. getNextReviewTimeInfo메서드 호출 후에 리스트에 각각 새로운 id와 date를 추가하여 saveNextReviewTimeInfo를 호출하여 직렬 화된 파일인 BrainAlarm.data를 저장하는 식으로 구현되어 있다.

2. getNextReviewTimeInfo 메서드

매개변수로 Context context, ArrayList<Integer> idList, ArrayList<Long> dateList를 받으며 idList와 dateList에다 현재 저장된 직렬 화된 파일에서의 int, long 리스트를 추가하는 동작을 하게 구현되어 있다.

3. deleteOneNextReivewTimeInfo 메서드

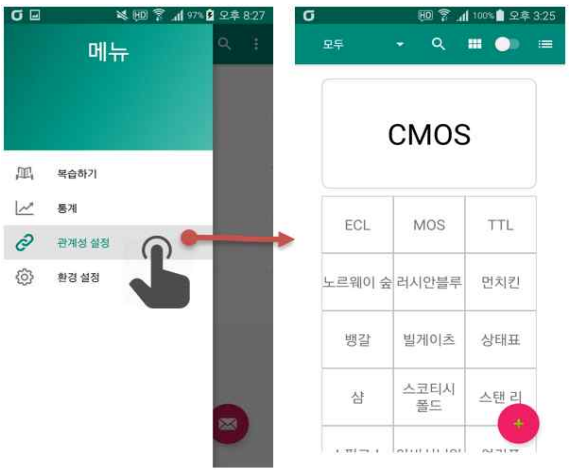
매개변수로 Context context, int id를 받으며
매개변수의 id는 현재 저장된 직렬 화된 파일에서 삭제할 키워드의 id이다.
이 메서드가 호출되면 해당 키워드의 직렬 화된 파일을 불러와서 각각의 리스트에

서 id와 그에 해당하는 dateList의 아이템을 삭제하고 다시 직렬 화하여 파일을 저장한다.

관계성 설정 기능

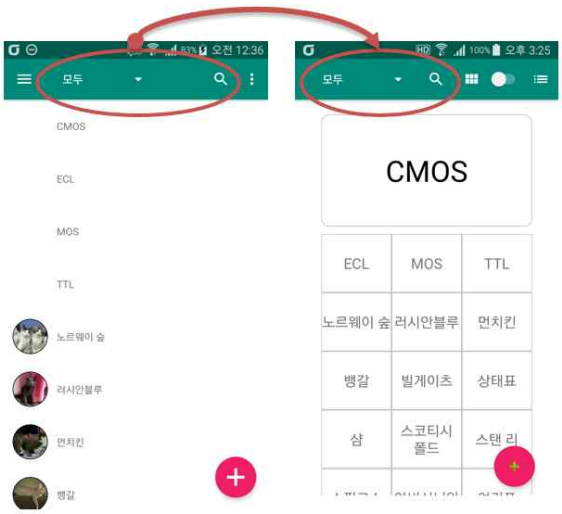
관계성 설정 기능의 설명은 사용자가 각 키워드의 관계성을 추가하고 싶을 때 메인메뉴 내비게이션에서 관계성 설정을 눌렸을 때의 MainActivity 화면 컴포넌트 객체와 RelationActivity 화면 컴포넌트 객체 간의 동작을 기술한다.

1. MainActivity의 내비게이션에서 '관계성 설정' 터치했을 때의 동작



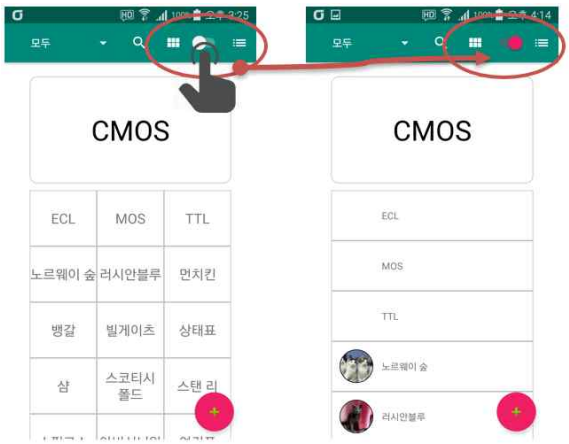
이 이벤트가 발생하면 MainActivity에서 현재 보고 있는 카테고리의 키워드들을 Intent에 id를 기반으로 ArrayList<Integer>에 저장하여 직렬 화하여 전달한다.

RelationActivity는 AppCompatActivity를 상속하는 Acitivity 컴포넌트이다.
이 RelationActivity 컴포넌트의 클래스는 내부 클래스로 RecyclerView.Adapter를 상속하는 클래스인 RelationGridRecyclerAdapter와 RelationListRecyclerAdapter를 가지고 있다.



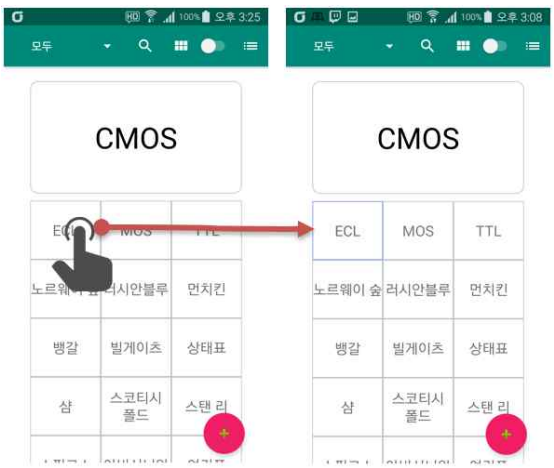
RelationActivity는 위와 같이 메인화면에서 사용하던 앱바(Appbar)의 카테고리와 검색 아이템을 재사용 하고 있으며 관계성 설정 화면에서도 카테고리 별로 보기 혹은 검색을 지원한다.

2. 보기방식 변경 스위치를 토글 했을 경우의 동작.



RelationGridRecyclerAdapter과 RelationListRecyclerAdapter를 사용하는 내부 레이아웃을 스위치 토글을 통해 변경한다.
RelationGridRecyclerAdapter가 처음 '관계성 설정'버튼을 눌렸을 때 보여 지는 어댑터 레이아웃이고 이는 현재 카테고리에 맞게 여러 키워드들의 이름을 Grid형태(직사각형 격자)로 사용자에게 제공한다.
RelationListRecyclerAdapter는 MainActivity의 어댑터 클래스를 재사용한 것으로 일부 필요 없는 메서드들의 동작을 재정의 하고 있다.

3. 나열되는 키워드들을 터치했을 때의 동작.



나열된 키워드들을 터치하면 위와 같이 동적으로 터치 되었다는 것을 표시하도록 아이템 Keyword 선택여부(select필드)와 추가·삭제된 관계성을 DB의 관계성 테이블에 즉시 갱신하고 화면에 보여 지는 아이템의 레이아웃을 변경한다.

4. FloatingAction 버튼을 터치했을 때의 동작

내부적으로 타겟 키워드 리스트들의 현재 키워드 인덱스를 1 증가시켜
다음 키워드의 관계성을 설정할 수 있게 한다.
SnackBar 메시지를 통해 사용자에게 몇 개의 관계성이 추가되었는지 보여준다.

통계화면의 기능

1. 통계 액티비티

임시적으로 스피너로 구성, 스피너에서 15일, 한 달 중 선택하게 되면 해당 기간의 그래프로 이동한다.

2. 그래프 액티비티

사용자의 문제풀이시간과 정답률에 관한 그래프들로 구성되어있다. 해당 그래프들은 오픈소스 라이브러리인 mpAndroidChart(그래프의 인자 값만 입력하게 되면 자동으로 x축과 y축을 구성해서 그래프로 표현)를 이용해서 제작하였고, 스피너를 이용해서 각각 그래프의 y축 목표치를 설정하여 사용자가 자신의 학습 능률을 쉽게 파악하고 목표치를 설정하도록 구성하였다.

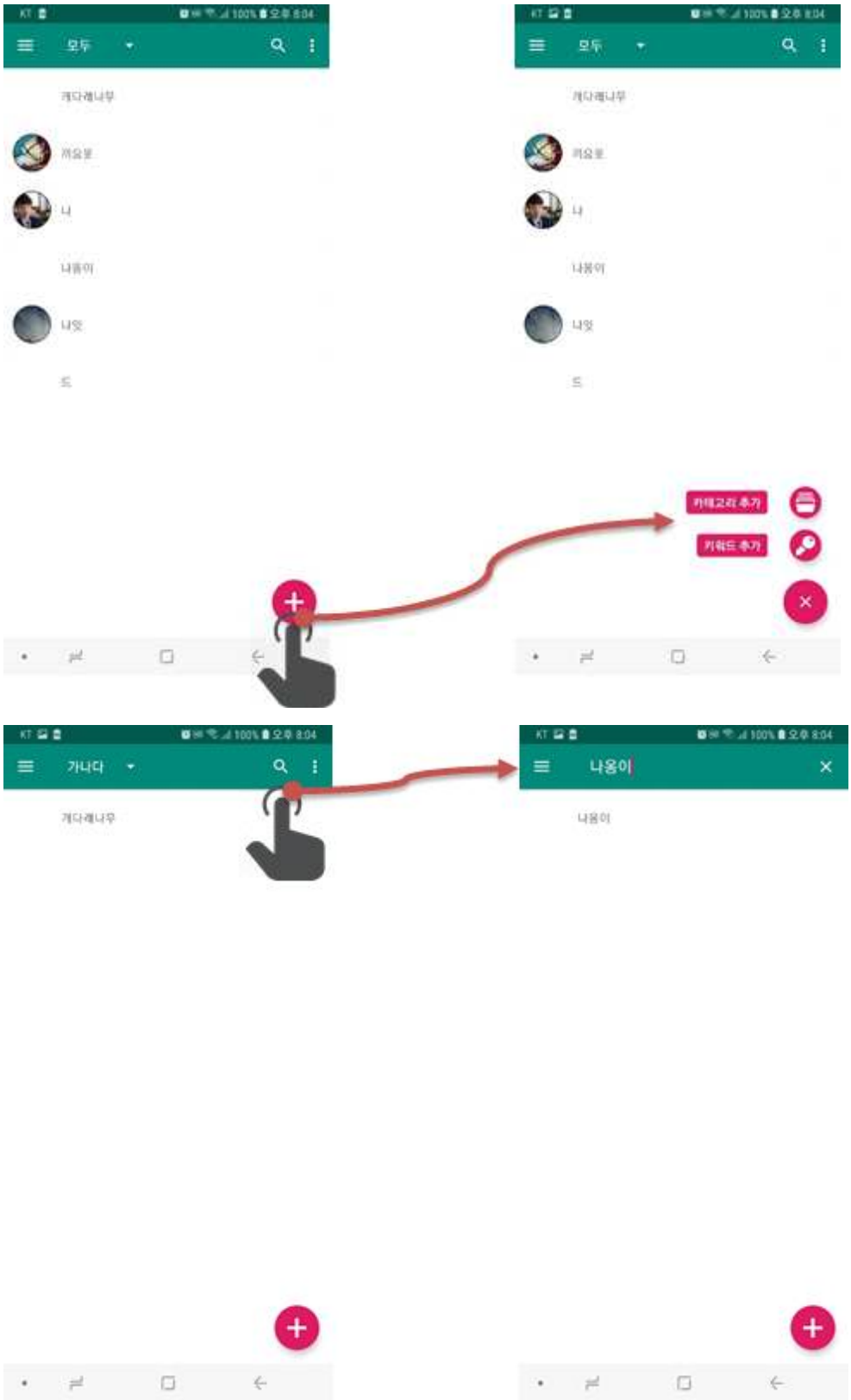
설정화면의 기능

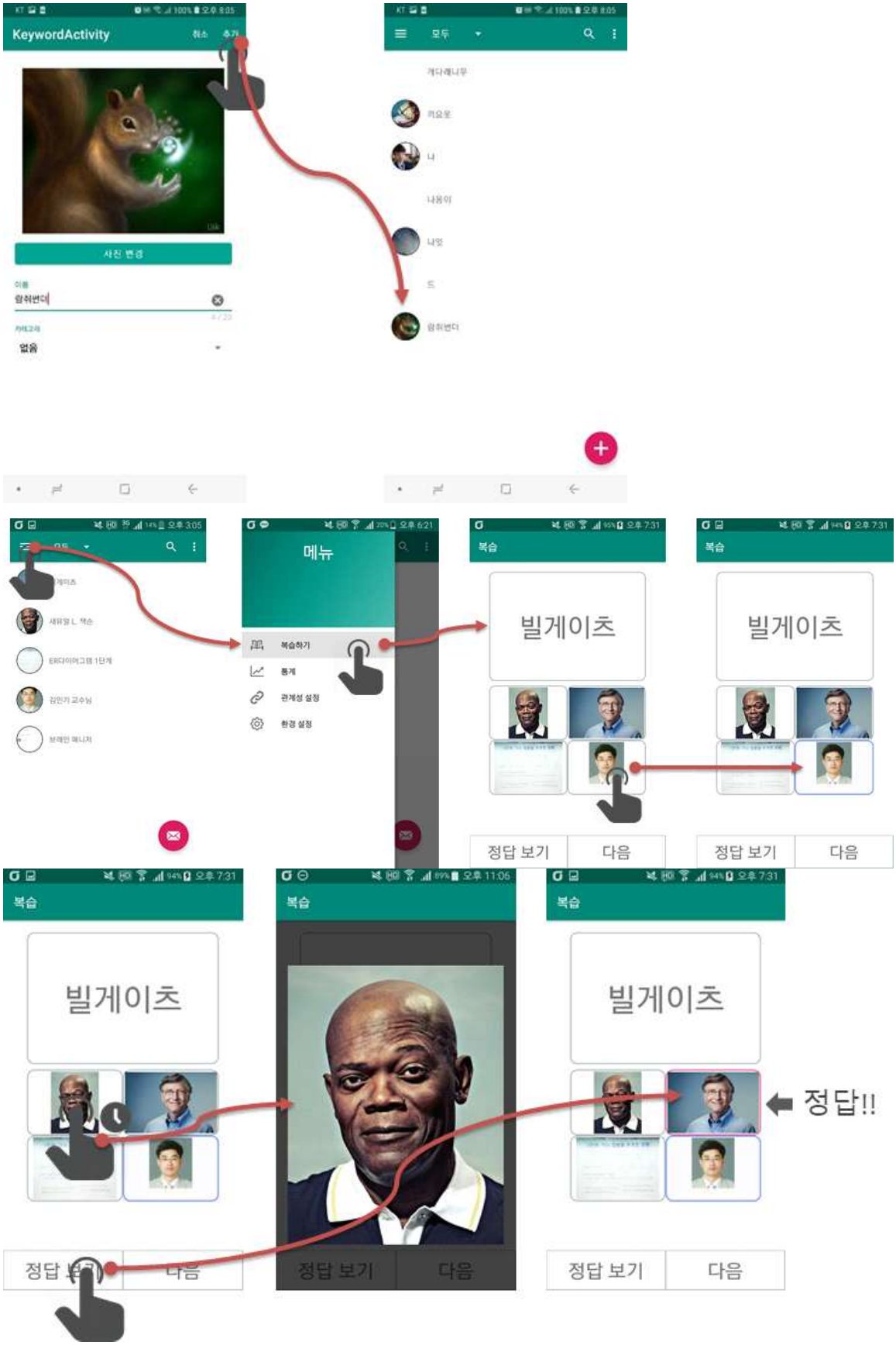
설정화면은 사용자가 앱을 사용하는데 있어 여러 가지 설정정보를 저장하고 보여주는 역할을 한다. 이곳에서 학습형태지정, 야간모드, 이미지 등록 시 원본이미지 삭제 옵션을 선택할 수 있고, 개인정보 처리방침과 오픈소스 라이선스를 확인 가능하다.
사용자의 망각곡선을 보여주고, 해당 곡선을 커스텀 하여 반복주기를 사용자가 조정할 수 있게 할 계획이었으나, 시간관계상 해당 기능은 미구현 상태이다. 그 외, 학습 알림 형태 중 알람과 잠금 화면도 구현하지 못하였다.

개인정보 처리방침과 오픈소스 라이선스

해당 기능은 실시간 업데이트가 되어야하고 많은 양의 텍스트 리소스가 필요한 특성상, 웹뷰어를 이용하여 구성하였다. 웹뷰 액티비티에 간단한 웹뷰어를 구성하고 URL을 넘겨받아 페이지를 참조한다. 각 페이지는 깃헙 정적 페이지로 구성되었으며, 파일은 html형식과 md형식으로 작성하였다.
(주소 : https://neomindstd.github.io/app_docs/privacy_policies/BrainManager, https://neomindstd.github.io/app_docs/opensrc_libraries/BrainManager)

3.3 UI 구성




KeywordActivity
취소
추가
사진 변경
이름
암취번데
카테고리
없음
모두
메뉴
복습하기
통계
관계성 설정
환경 설정
복습
빌게이츠
정답 보기
다음
정답!!

3.4 데이터 구성

구분	용량 (단위 : MB)
사진 및 그림 데이터(.xml(svg), png)	4.4
프로젝트 용량	156.07
배포 용량(.aab, Android App Bundle)	3.85
다운로드 용량(.apk)	5.52~5.54(기기별로 상이함)
설치 시 용량	8.04~17.9(기기별로 상이함)

4. 구현 세부사항

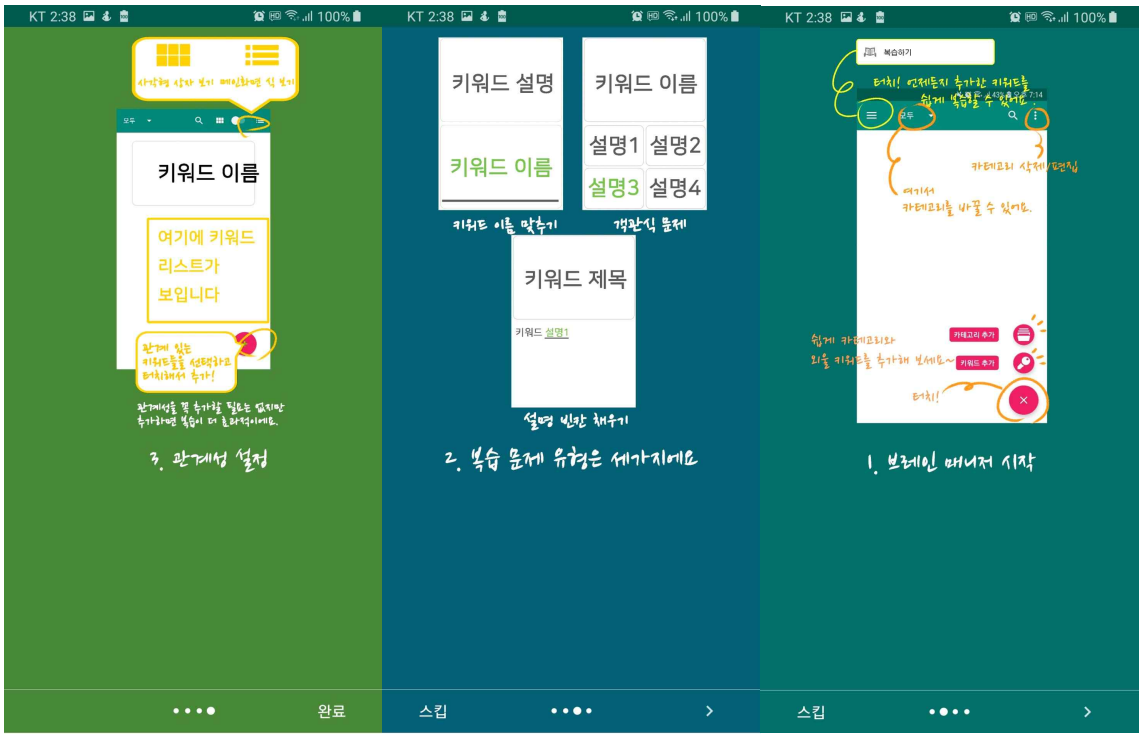
이하 파일경로는 기본 패키지 디렉토리(BrainManager₩app₩src₩main₩java₩std₩neomind₩brainmanager₩) 이하 경로이다.

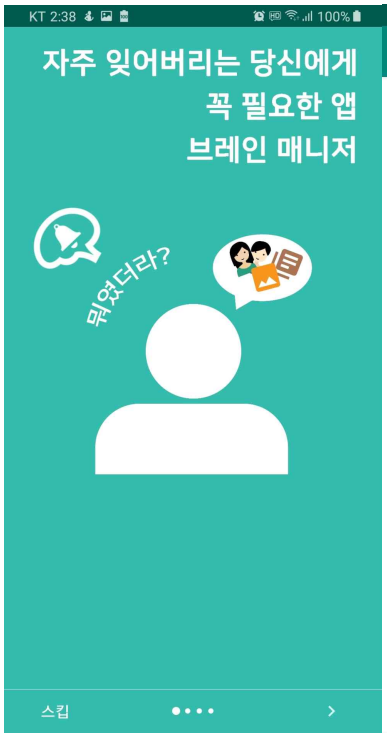
4.1 기능 구현 (파일명 / 클래스명)

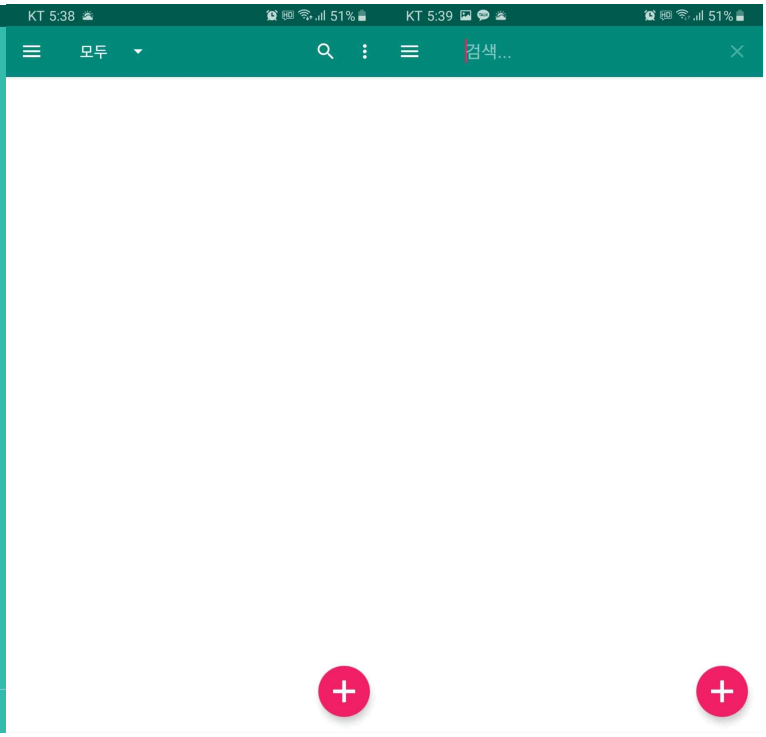
- 알람 관리 : utils₩AlarmReceiver.java / AlarmReceiver
- 데이터 베이스 관리 : utils₩BrainDBHandler.java / BrainDBHandler
- 객체 직렬 화 : utils₩BrainSerialDataIO.java / BrainSerialDataIO
- 저장소 파일 관리 : utils₩FileManager.java / FileManager
- 알람 : utils₩NotificationReceiver.java / NotificationReceiver
- 권한 관리 : utils₩PermissionManager.java / PermissionManager

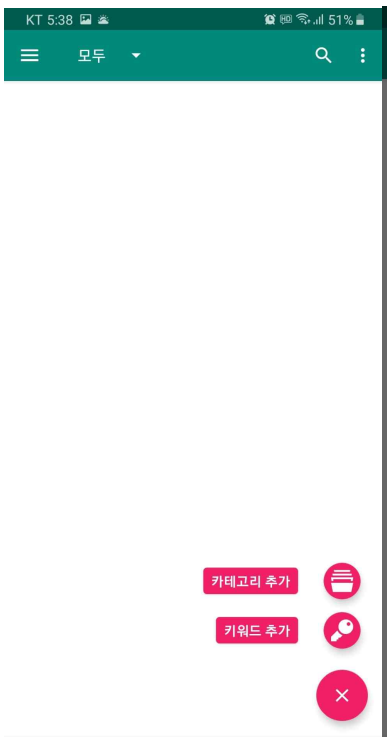
4.2 UI 구현 (파일명 / 클래스명)

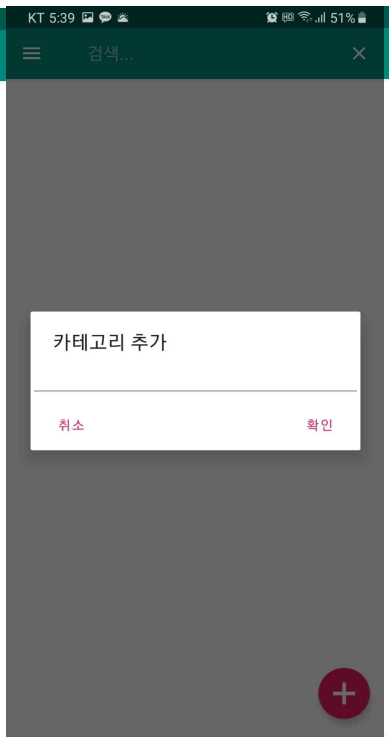
파일명	MainActivity.java
클래스명	MainActivity



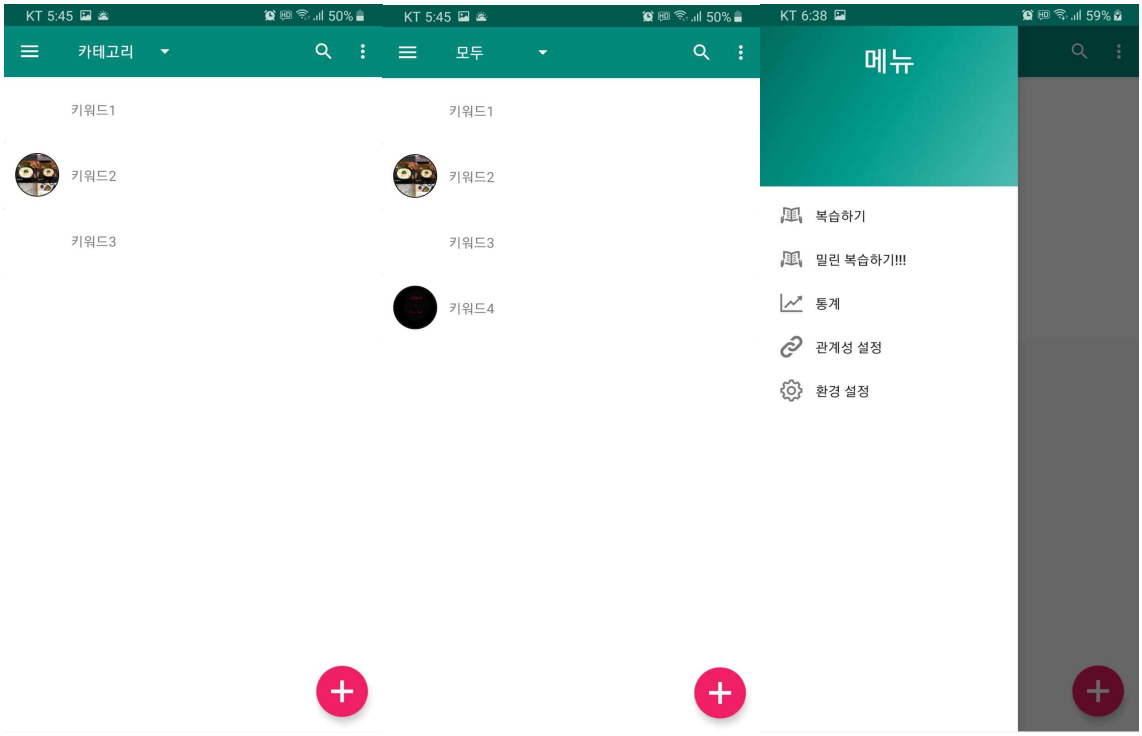

KT 2:38
100%
자주 잊어버리는 당신에게
꼭 필요한 앱
브레인 매니저
뭐였더라?
스킵


KT 5:38
51%
모두
KT 5:39
51%
검색...

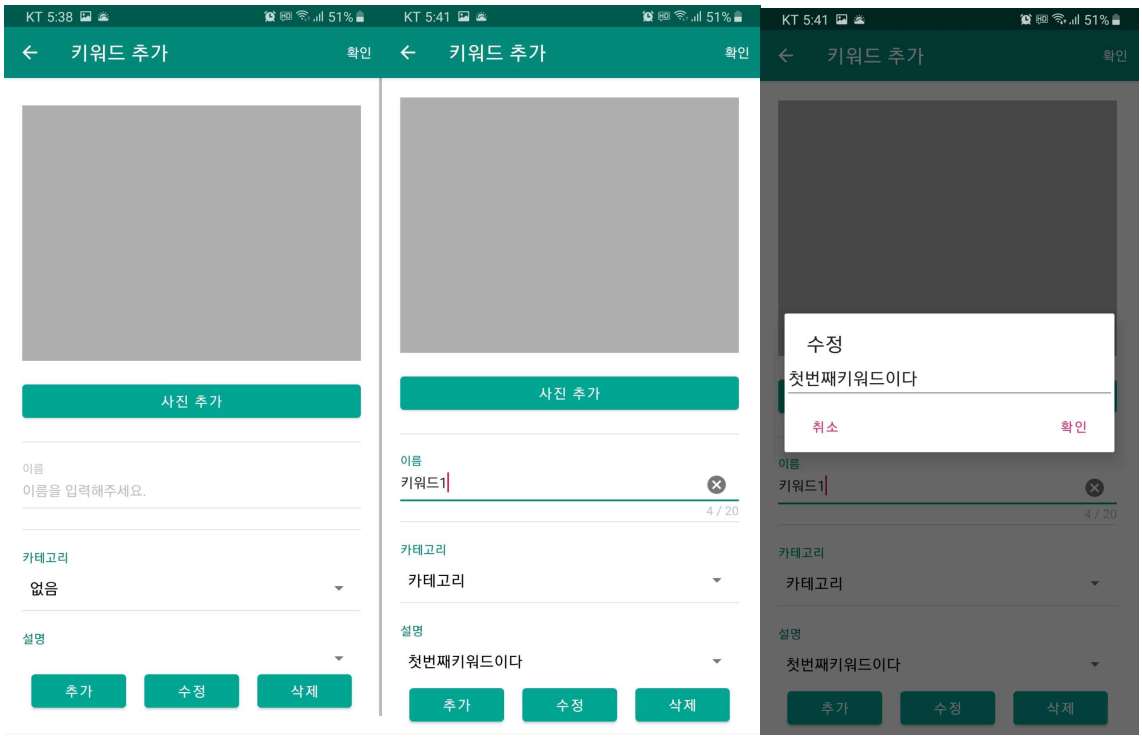

KT 5:38
51%
모두
카테고리 추가
키워드 추가

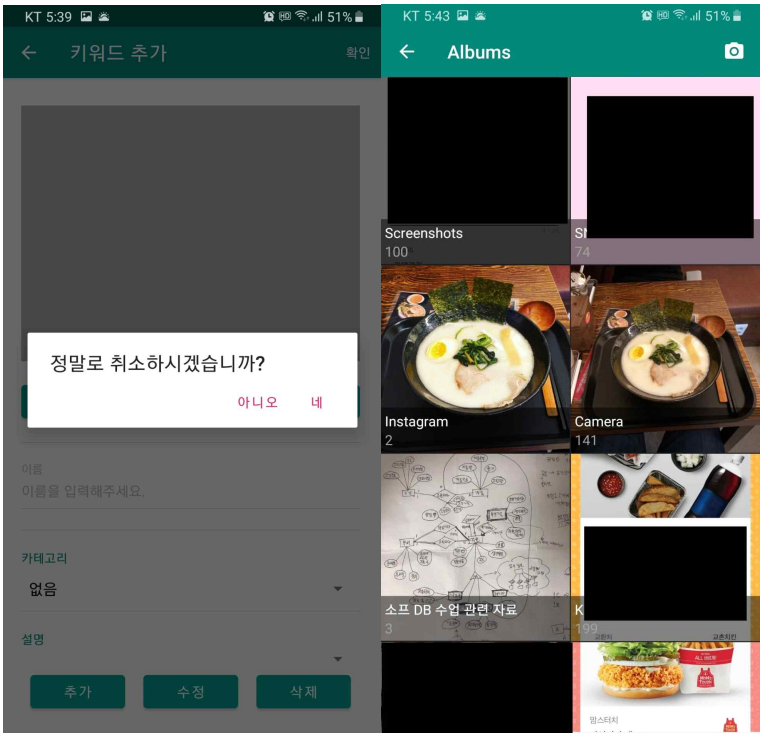

KT 5:39
51%
검색...
카테고리 추가
취소
확인


KT 5:39
51%
카테고리
현재 카테고리 삭제
현재 카테고리 편집

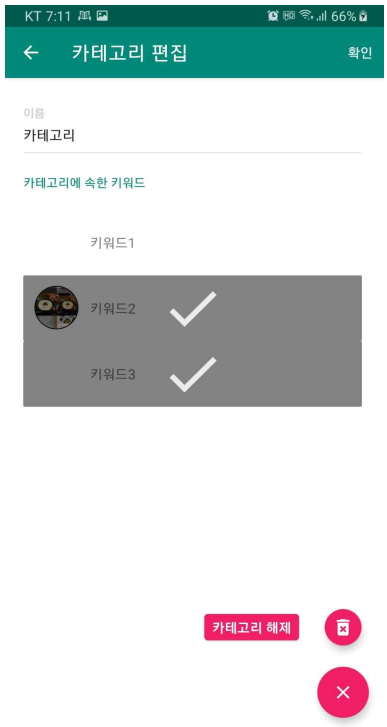


파일명	KeywordActivity.java
클래스명	KeywordActivity

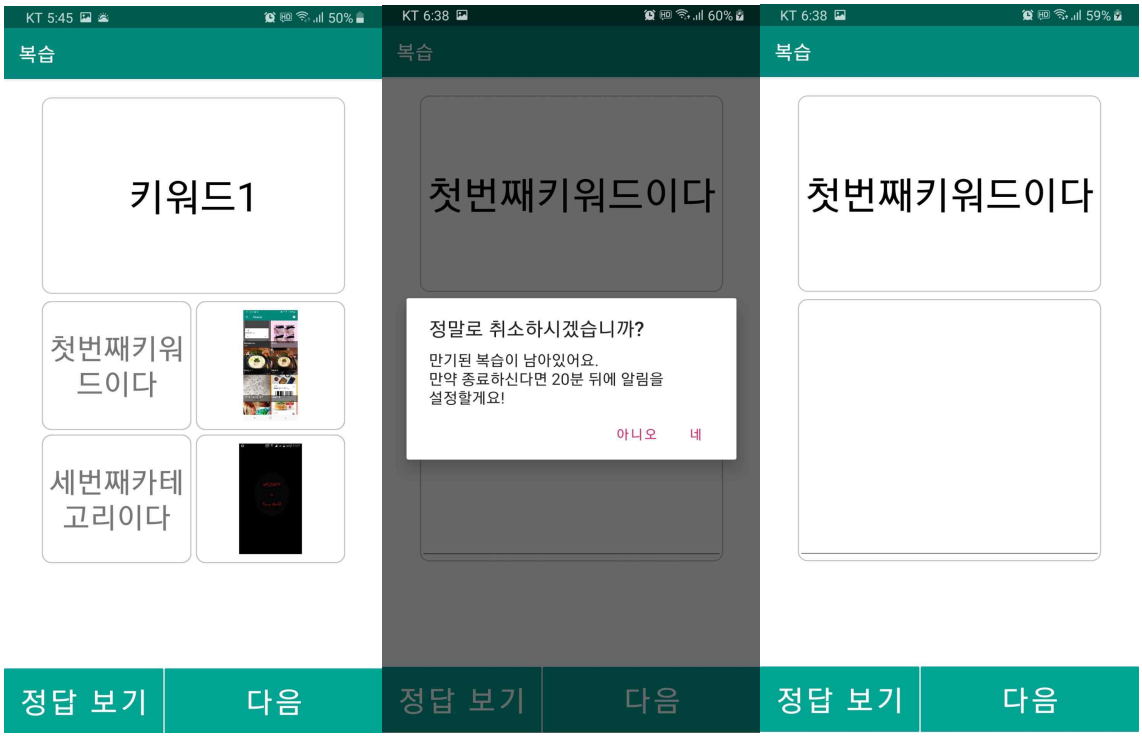




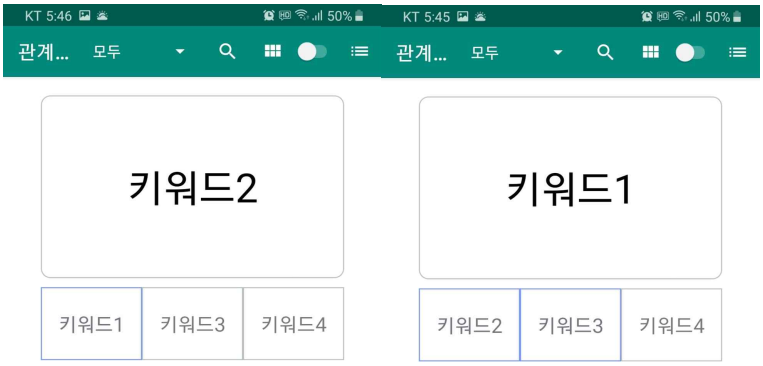
파일명	CategoryActivity.java
클래스명	CategoryActivity

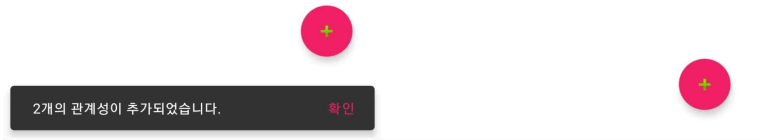


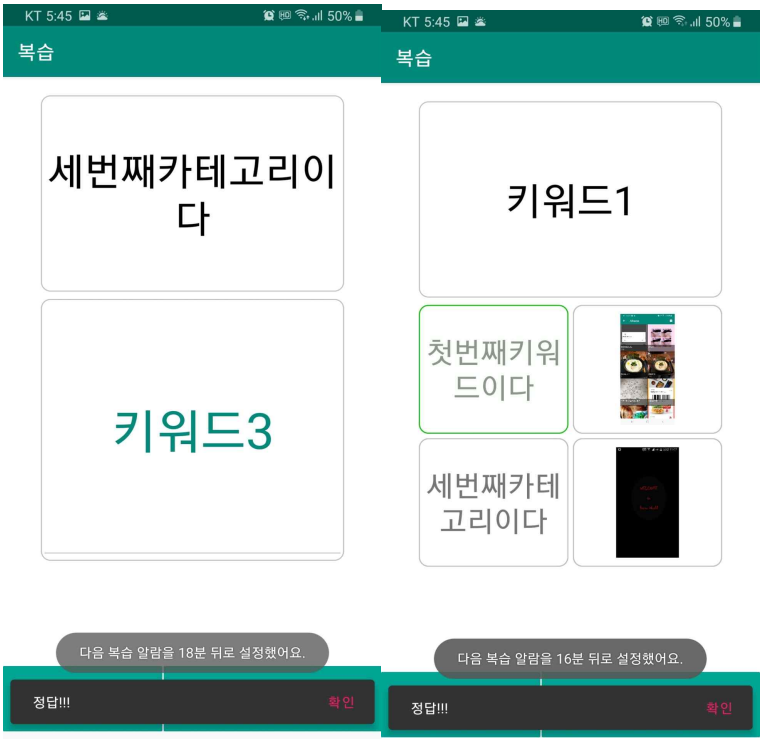
파일명	ReviewActivity
클래스명	ReviewActivity



파일명	RelationActivity
클래스명	RelationActivity

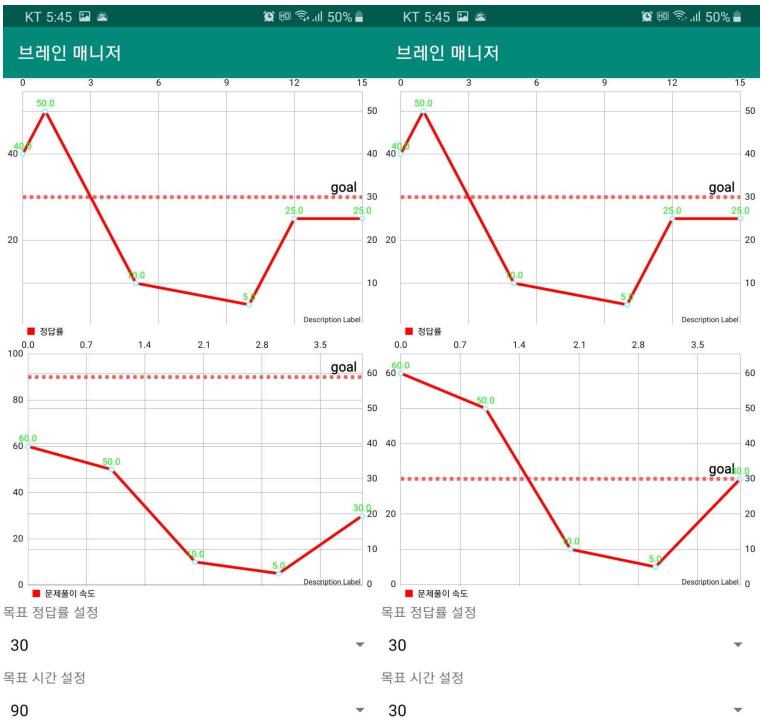




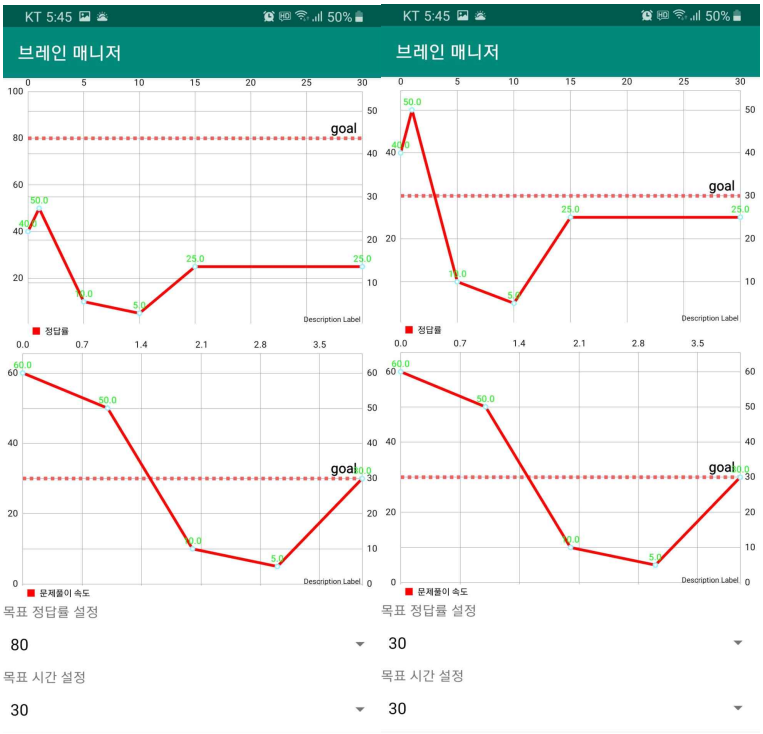


파일명	StatisticsActivity
클래스명	StatisticsActivity

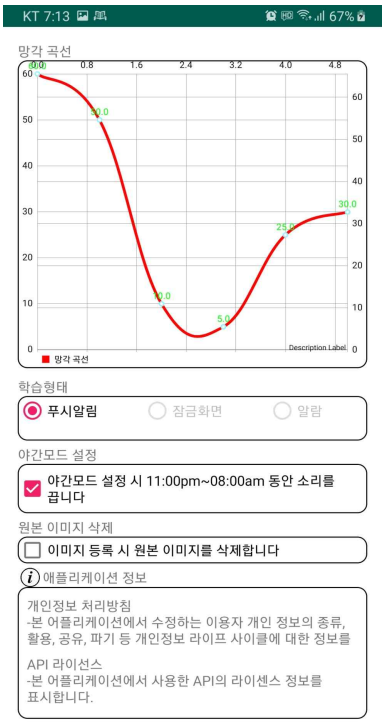
파일명	FifteenDaysChartActivity.java
클래스명	FifteenDaysChartActivity



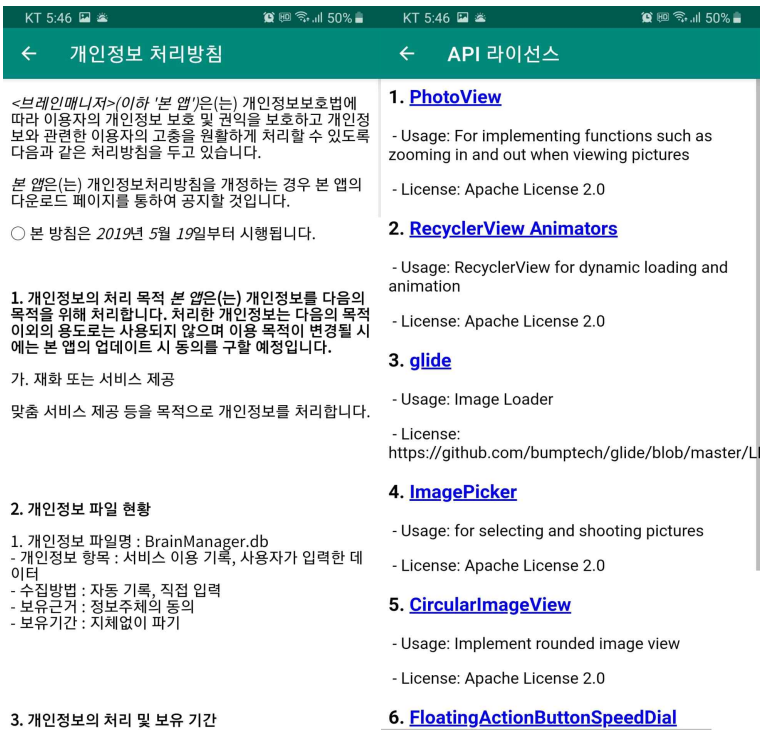
파일명	OneMonthDaysChartActivity
클래스명	OneMonthDaysChartActivity



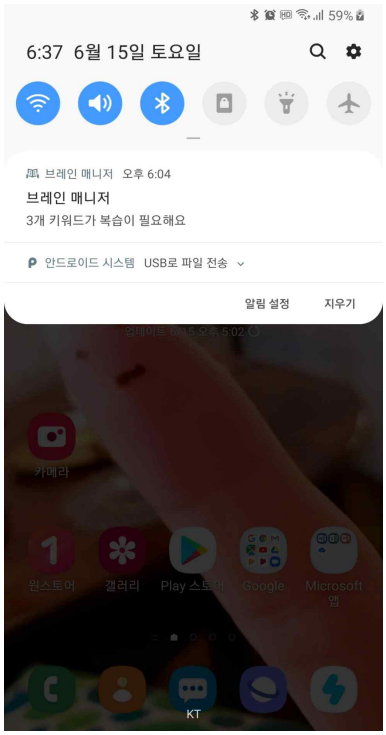
파일명	SettingsActivity
클래스명	SettingsActivity



파일명	WebViewActivity
클래스명	WebViewActivity



알람 동작 시



앱 아이콘

4.3 데이터 구축 및 관리 (파일명 / 클래스명)

- 카테고리 : data₩Category.java / Category
- 설명 : data₩Description.java / Description
- 키워드 : data₩Keyword.java / Keyword
- 학습 : data₩Test.java / Test

4.4 테스팅 및 품질평가 방법

기본적으로 하나의 모듈을 만들 때 마다 개발자 각각 단위테스트와 동적 화이트박스 테스트를 실시하였다. 모듈을 병합 후 통합 테스트와 시스템 테스트를 진행하였고 앱을 플레이 스토어에 출시한 후로 google firebase의 robo 테스트에 의해 동적 블랙박스 테스트 도움을 받을 수 있었다.

	APK	오류	경고	사소한 문제	완료된 테스트	보고서 생성 날짜
⚠	19	0	390	273	10	2019. 6. 16. 오후 6:02:33
⚠	18	0	295	164	10	2019. 6. 16. 오후 5:48:38
⚠	17	0	181	69	10	2019. 6. 16. 오후 2:05:37
⚠	16	0	283	97	10	2019. 6. 5. 오후 4:06:16
⚠	15	0	452	331	12	2019. 6. 4. 오후 9:08:09
⚠	14	0	295	225	12	2019. 6. 4. 오후 12:52:24
⚠	13	0	269	152	12	2019. 6. 3. 오후 6:39:43
⚠	12	0	330	193	12	2019. 6. 3. 오전 2:03:23
!	11	1	244	145	12	2019. 6. 3. 오전 1:36:31
⚠	10	0	196	113	12	2019. 6. 3. 오전 12:17:48
⚠	9	0	231	101	12	2019. 6. 2. 오전 1:01:21
⚠	8	0	435	229	12	2019. 6. 1. 오후 7:14:28
!	7	6	271	147	12	2019. 6. 1. 오후 4:35:11
!	6	7	178	218	12	2019. 6. 1. 오후 1:38:19
!	5	4	176	41	12	2019. 5. 31. 오후 11:43:21
!	4	4	272	192	12	2019. 5. 31. 오후 3:02:35
!	3	4	263	236	12	2019. 5. 30. 오후 2:17:34
!	2	3	134	101	12	2019. 5. 27. 오전 9:39:16
!	1	8	64	30	12	2019. 5. 26. 오후 10:05:08

사진과 같이 도움을 받을 수 있었고, 오류가 나던 상황의 logcat과 테스트 영상, 각 기기별 화면 상태 등 다양한 정보를 제공해준 덕분에 수월하게 테스트 받을 수 있었다.

또한 다른 사람들에게 앱을 알려주고 써보라고 한 후 일반 사용자 대상으로 동적 블랙박스 테스트를 받을 수 있었고, 최종 발표 이후로는 다른 조의 조원들도 테스트하고 의견을 받을 수 있었다.

앱의 규모가 어느 정도 있기 때문에 비용과 시간을 고려하여 경로 또는 노드 커버리지 테스트는 하지 못했지만, 많은 테스트와 버그 수정을 통해 실사용에 문제가 없는 정도로의 테스트는 완료하였다.

4.5 SW 설치 및 사용법

1. 구글 플레이 스토어에서 브레인 매니저 검색 => 허용된 국가에서 호환되는 기기, 안드로이드 버전이면 검색이 된다. => 설치

1-1. 또는, 동봉된 cd의 설치파일(.apk)를 이용하여 설치하거나 직접 프로젝트를 빌드하여 설치할 수 있다. 안드로이드 기기가 없는 경우 에뮬레이터를 사용할 수도 있다.

2. 튜토리얼 화면에서 시작

3. 튜토리얼 숙지 후 사용자는 오른쪽 하단의 버튼을 통해 외울 키워드를 추가함.

4. 추가하는 순간에 학습을 한번 한 것으로 간주하고 자동으로 20분 뒤에 복습 알림이 울리게 등록됨.

5. 세 가지 경우 있다.

①사용자는 별도의 추가 동작 없이도 복습 알림만으로 망각곡선에 따라 잊어버리지 않게 복습이 가능하다.

②복습 알림이 나타나기 이전에 사용자가 임의로 복습하기를 눌려서 복습한 경우 => 이미 등록된 복습 알림에는 영향을 주지 않고 기억용이성(망각곡선 알고리즘 변수) 수치만 변하며 사용자는 중간 검정을 할 수 있다.(원래라면 통계보기로 이걸 확인할 수 있어야함.)

③복습 알림이 왔는데 취소했거나 복습알림 시간이 지난 키워드들이 있음에도 복습을 하다가 중간에 중지한 경우=>기본적으로 20분 뒤로 다시 복습 알림이 등록되며 메인화면 내비게이션 슬라이드에 밀린 복습하기 항목이 추가로 생겨서 복습알림 시간이 지난 키워드들만을 복습할 수 있다.

①②③모든 경우에 대하여 복습을 한 경우 알고리즘에 따라 복습간격이 정해지고 이에 따라 다음 복습 알림이 지정된다.

5. 프로젝트 성과, 문제점 및 활용 방안

5.1 프로젝트 성과

도우진

작년 기초설계 및 프로젝트와 웹 프로젝트에 이어 조장으로 진행해본 프로젝트입니다. 원하는 사람끼리 구성된 팀이 아닌, 랜덤으로 팀을 구성한 이번 프로젝트를 통해, 많은 것을 얻을 수 있었습니다.

팀원 각자의 능력이 다를 때 업무 배분을 어떻게 해야 하는지, 팀원끼리의 의견 교환은 어떻게 해야 할 지에 대해 많은 생각을 하게 됐습니다. 기존에는 회의는 최소화하고 실질적인 개발이 중요하다 느껴 이번 프로젝트에서도 같은 방침을 적용했었습니다. 하지만, 의견교환이 생각보다 제대로 이루어지지 않았고(하늘 씨와 정민 씨의 파트가 db에 연동하여 망각곡선을 변경하고 키워드 알림은 db에 저장된 값을 통해 설정되어야 했습니다. 알림 부분은 장예찬 팀원의 도움으로 해결하였습니다.) 팀원 각자의 진행사항을 파악한 것이 실제와 괴리가 꽤 있었습니다. 회의의 중요성을 통감할 수 있는 프로젝트였습니다. 최종발표에 가까워지던 시점까지도 결과물은 최악이라고 할 수 있을 정도였습니다. 하지만, 이내 문제가 있다는 것을 깨닫고 저희 팀이 프로젝트에 더 적극적으로 참여할 수 있게 유도하여 어느 정도는 완성할 수 있었습니다.

장예찬

우선 가장 힘들었던 것은 프로젝트에서 반드시 제대로 동작하는 것이 필요한 기능을 팀원이 구현하였는데, 이것이 한계라고 하여 요구사항과 전혀 안 맞는 코드를 최종발표 2일 전에 받게 되어 전해 받은 시점이 너무 늦은 것이 가장 큰 문제였고 이것이 제일 처리하기 힘들었습니다. 원래라면 이미 기능 병합 후 테스팅과 사용자 UX를 개선이 완료된 상태로 영상물을 제작해야 하는 기간이었습니다. 어쩔 수 없이 요구사항에 전혀 맞지 않는 팀원의 코드를 급하게 가용 가능한 인원인 제가 이틀 밤낮을 새며 재작성하여 요구사항을 어느 정도 만족시킬 수 있었습니다. 당연히 그만큼 최종 테스팅 기간은 짧아졌고 UX는 개선할 수 없었으며 그나마 조원들을 다독여 ppt와 영상물을 제작하도록 시켰지만 짧은 기간이여서 질도 나빠졌습니다. 변명이라면 변명이지만 발표 시점에도 저희 팀은 대본도 없고 저는 이틀을 밤새고 3시간을 자다가 발표를 하러오게 되어 제가 완성도 높게 구현했던 내용들도 청자들에게 와 닿게 설명하지 못해서 너무 안타까웠습니다. 발표가 27일이었는데 이 발표를 29일로 미루지 못한 것이 아쉬웠고 29일까지는 재작성한 기능들에 대해서 완전히 테스팅과 UX점검을 마쳐서 최종작품의 컴퓨터과학과 학우들의 테스팅과 댓글에서 좋은 결과를 얻을 수 있게 되었습니다.

이런 일이 생긴 이유는 저와 조장은 팀원들이 아무 얘기가 없어서 예정 계획대로 잘 진행되고 있을 것이라 생각하였고 병합은 안됐어도 각자라도 단위 테스트를 수

행했습니다. 나머지 팀원의 기능구현 및 병합이 예정 계획 오차를 너무 벗어나서 확인해보니 나머지 모두 예정대로 수행하지 못하고 있던 상황이란 것을 상당히 늦게 인지하게 되었습니다. 그제야 우리 팀이 팀원 간에 의사소통을 너무 소홀히 했다는 것을 알게 되었지만 이미 늦은 후였습니다.
이번 프로젝트에선 저와 가까운 관계인 도우진 조장 외 팀원들의 역량을 정확히 알지 못하여 역할분장에서 요구사항을 너무 높은 수준으로 잡은 것 같고, 그나마 UX에 영향을 덜 끼치는 기능의 경우에는 요구사항을 만족하지 않아도 괜찮았지만 UX에 지대한 영향을 끼칠 수 있는 기능의 경우에는 요구사항을 만족하지 못했을 때 굉장히 큰 리스크가 있기 때문에 역할 분장을 신중히 정해야 된다는 것을 알게 되었습니다.

성과에 대해 기술하자면 우선 제 업무분장 요구사항에서 상당히 난이도가 높은 과제들이었음에도 목표한 만큼의 구현을 100% 완료했다는 것이 가장 만족스럽습니다. 프로젝트의 핵심기능을 위해 여러 간격반복 알고리즘과 논문들을 열람했는데 전부 영어로 작성되어 있어 이를 해독하는 것이 학업에 유익했고
이들 중 가장 간단하고 강력한 알고리즘, 그리고 특허 문제가 없는 알고리즘을 선별하고 유의미 학습에서 영감을 받아 이 알고리즘을 새롭게 재구성하는 것이 굉장히 유익했습니다. 또한 유의미 학습에서 추가로 영감을 받은 관계성 기반 정렬과 같은 새로운 정렬 알고리즘 개발도 논리적인 사고력을 기르는데 도움이 되었습니다. 여러 프로그래밍 프로젝트의 다양한 경험이 있었던 것이 이번 프로젝트를 하는데 도움이 많이 되었습니다.
프로젝트를 진행하는 중에 처음 작성했던 제 요구사항에서 중간 테스트들, 최종 댓글로 인해 사용자 경험을 위해 추가적인 요구사항들이 발생하게 되었는데 그것들도 모두 깔끔하게 완료해서 좋았습니다.
여러 오픈소스 라이브러리들을 검토하고 사용하면서 깃허브 라이브러리 사용법을 익혔고 깃허브를 사용법을 처음 배웠지만 프로젝트 중반부턴 완벽하게 마스터해서 앞으로의 프로젝트에서 도움이 많이 될 것 같습니다.
추가적으로 저에게 예정되지 않았던 다른 파트도 맡게 되어 많이 힘들었지만 더 폭 넓은 프로젝트를 진행할 수 있게 되어 좋았고 이번 프로젝트로 얻은 여러 경험들은 이후에 긍정적으로 작용할 것 같습니다.
전체적으로 이번 프로젝트에서 제 성과는 굉장히 만족스럽고 다음 종합 설계 프로젝트 강의가 기대됩니다.

김정민

프로젝트를 진행하면서 안드로이드 개발에 대한 기본적인 틀을 체득 할 수 있었고 외부 라이브러리 등을 활용하는 법에 대해서 배우게 되었습니다.
소프트웨어 개발에 대해 막연히 생각만 해봤었으나 이번 프로젝트 수행으로 인해

하나의 소프트웨어를 완성하는데 상당한 시간이 소요되고 소프트웨어를 구현하기 위해 배워야 할 것이 많다는 것을 느낀 것 같습니다.

하늘

자바와 안드로이드 관련해서 아는 게 아무것도 없어서 자기 주도적으로 학습하면서 진행했기 때문에 제대로 잘하였느냐보다 스스로 공부하면서 진행하였다는 것에 뿌듯함을 느낍니다.

또한 깃 허브 사용 및 안드로이드 앱 개발에 대해 피부에 와 닿게 배울 수 있었습니다.

소프트웨어 개발에 있어 제 실력이 많이 부족하다는 것을 느꼈고, 계획을 아무리 잘 짜더라도 그대로 수행한다는 것은 어려운 일이란 것을 실감했습니다.

이로 인해 배우고 공부하며 더욱 성장하는 계기가 된다면 더할 나위 없이 좋지 않나 싶습니다.

마지막으로 이제 저는 그 작은 앱 하나에도 많은 사람의 생각과 시간과 노동이 들어가 있다는 걸 깨달았기 때문에 조금 부족한 앱에 대해서도 함부로 나무라지 않을 것 같습니다.

5.2 문제점 및 활용 방안

5.2.2 활용 방안

사용자의 기억을 체계적인 방식으로 관리 가능하고, 망각 곡선과 유의미 학습을 활용한 관계성 학습을 통해 효과적으로 기억을 장기 기억으로 유지하며, 사용자의 학습에 큰 도움을 줄 수 있을 것으로 보입니다.

설명엔 텍스트와 이미지까지 추가가 가능하기 때문에 인물의 사진을 통해 사람을 기억한다거나, 장소의 사진으로 장소를 기억하는 데 활용할 수 있습니다. 또, 복수개의 설명을 추가할 수 있으므로 복수의 설명이 필요한 이론, 영단어 등을 학습하는 데 활용할 수 있습니다.

등록한 키워드들은 자동으로 망각곡선+관계성 알고리즘에 의해 잊어버릴 쯤의 시기에 자동으로 사용자에게 푸시알림(배너+소리+진동)으로 복습하라는 메시지를 보여줍니다.
이를 터치하면 간편하게 잊어버릴 시기가 된 키워드들에 대해 복습이 가능하므로 사용자는 등록된 키워드와 내용을 장기기억하기 더 쉬워지고 잊어버릴 염려가 줄어듭니다.

이는 실제로 학업을 위한 목적으로 시험에서의 활용도 가능한 것을 확인되었고 브레인 매니저의 자동 복습 관리를 사용해서 공부를 한 한생은 키워드들을 잊어버릴

수가 없었다고 설명했습니다.(최종 댓글, 다른 학교 테스터의 답변)

6. 참고 자료

Richard Gunstone. 2015. Meaningful Learning. Faculty of Education, Monash University, Clayton, VIC, Australia
: 유의미 학습에 대해 뇌 과학과 통계적 기법을 사용하여 그 유용성에 대해서 증명한 논문이며 주요내용은 관련된 것을(개념적 하위내용, 같이 배운 내용 등등) 동일 시기에 학습하거나 복습하면 뇌의 시냅스 그물 구조상 더 높은 학습 성취를 얻을 수 있다는 점, 관련된 내용들을 학습할 때 동일시기에 학습하거나 복습해야 이들의 다른 점을 구별할 수 있는 능력이 강화된다는 것을 설명하고 있습니다.

이 부분에서 영감을 받아 저희 앱의 망각곡선 알고리즘을 수정(관계성 있는 것으로 학습하면 학습이 더 잘된 것으로 판단)하고 관계성 기반 정렬(관계있는 것을 서로 인접하게 정렬) 구현했습니다.

Murre JMJ, Dros J. 2015. Replication and analysis of Ebbinghaus' forgetting curve. PLoS One. 10:e0120644.
(https://journals.plos.org/plosone/article/file?id=10.1371/journal.pone.0120644&type=printable)
: 제안서 작성 시에만 봤었던 논문입니다. 주요내용으론 실험을 하고 얻은 실험적인 데이터를 Python을 통해 가공하여 망각곡선의 주요 공식들을 증명하고 이를 좀 더 실험으로 나온 데이터에 맞게 변수를 추가하여 공식을 재가공한 하나의 지수함수를 보여주고 있습니다. 이 논문에서 망각곡선을 지수함수로 계산해낼 수 있다는 것을 알아내었습니다.

ZHAN, Jeff Hanksand Ping. The Forgetting Curve and Learning Algorithms.
(https://pdfs.semanticscholar.org/3eda/1f89d845a893cd4c8848b41697d39af19195.pdf)
: 설계서 작성 때, 위의 Murre의 논문만으로는 반복 간격을 얻어낼 수 없다는 것을 깨닫고 참고한 문서입니다. 주요 내용으로는 SuperMemo의 알고리즘인 SM-X 버전들의 상호 비교와 소개, 영어와 독일어에 대한 실험적 망각수치 데이터를 통한 모국어(영어)와 외국어(독일어) 단어의 기억효율을 비교한 것으로 실험결과는 더 기억하기 어려울 것이라 예상된 독일어보다 영어를 장기기억 하는 것이 더 어려웠다는 것을 보여줍니다.

뒷부분인 실험 내용은 저희에게 필요하지 않았지만
이 문서를 통해 재귀적 방식으로 지수 함수를 구현하여 망각곡선에 따른 복습

간격을 도출해낼 수 있다는 것을 알아내었고 SuperMemo의 알고리즘들을 서로 비교할 수 있었습니다. 이 알고리즘들 중 가장 근본이 되는 알고리즘이자 특허에 전혀 문제가 되지 않는 SM-2 알고리즘을 참고하여 수정하여 망각곡선 알고리즘을 제작하였고 주요 변경 점은 관계성 개념 추가와 조건 축소입니다.

SuperMemo SM-2 알고리즘
(https://www.supermemo.com/en/archives1990-2015/english/ol/sm2)
: 알고리즘 구현 시 참고했던 SM-2 알고리즘이 기술된 사이트입니다. 이를 저희 앱의 알고리즘 차별성에 맞게 개조하였습니다.

부 록

※ 제안서, 설계서, 발표자료(시연 동영상 포함), 최종보고서, 소스코드(데이터 포함)는 CD로 제출