Final Report: Team 11

[Mobile Programming]

N-Bang Travel

Travel Diary, Accounting book, Checklist

Dongwon Jeon 14146322

Soyoung Kim 15146307

Contents

1. Application Description

This application supports people who traveling with friends. It provides functions like travel diary, ledger for travel, checklist to memory and sharing on social media.

First of all, the main function of this app is accounting book that makes it easier for users to split traveling expenses with friends. User can create events, and record members and how much each one of them paid. This gives information about who need to pay to whom later. That makes it possible to settle money faster.

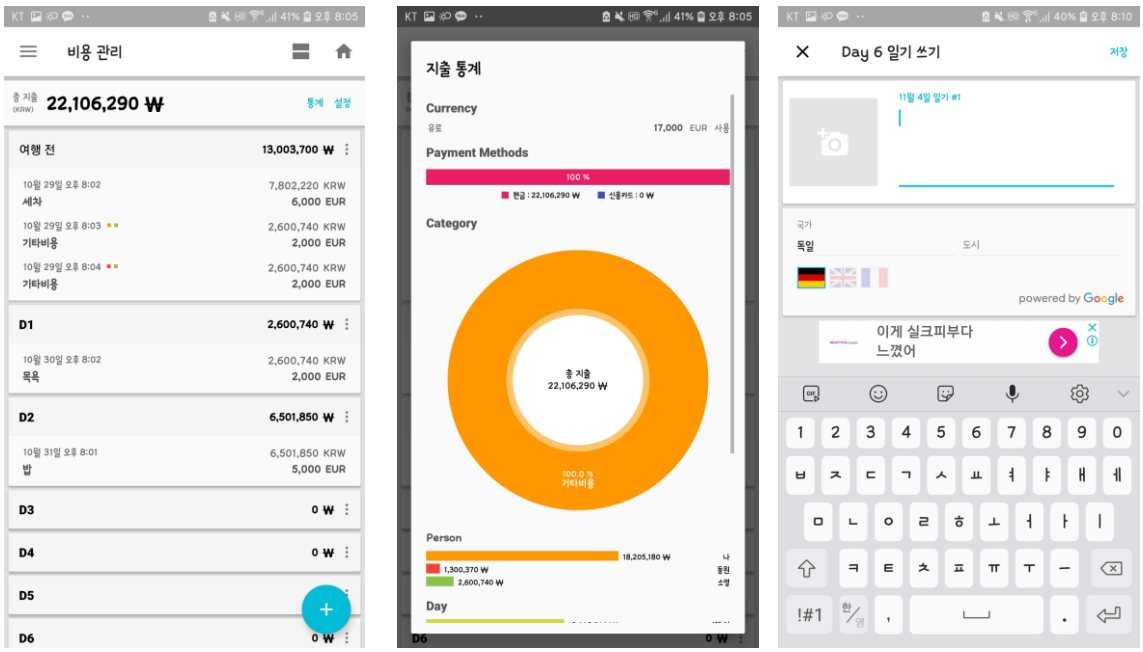
For example, suppose that travel members are ‘oney’, ‘twoy’, ‘threey’, ‘foury’. If oney spends $300 for bakery, and twoy spends $500 for coffee for all members, the app automatically calculates the money and supports the information “threey must give $200 to twoy, and foury must give $100 to oney and $100 to twoy”.

User can write diary in our app, and the diary can contain picture and brief documents. It also can share the diary to social media like Instagram, Facebook, or Twitter. In addition, the app provides essential checklist function for travel. The checklist function allows the user to write down what he or she needs to remember.

2. Comparing Existing Apps

	Travel App	Dutch Pay App	nBang Travel
Expenditure Record	O	X	O
Applying Exchange Rate	O	X	O
Checklist	O	X	O
Calculating Accounting Book	X	O	O
Sharing Accounting Book	X	O	O
Diary	O	X	O
Sharing Diary	X	X	O

2-1. Save Trip (Travel Application)



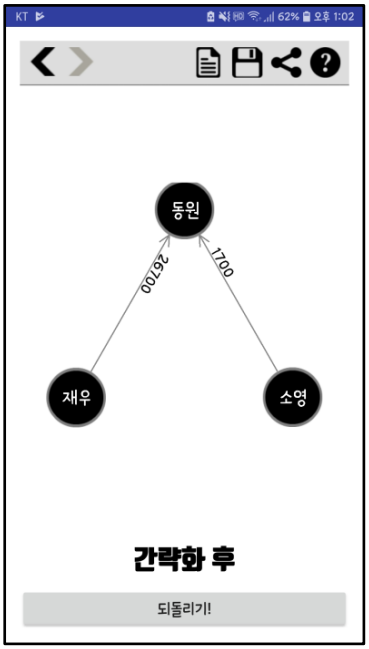
In this app, travel costs are categorized by date, and the statistics feature shows statistics on spending. Total expenditures can be divided by category or members. Travel diaries can add countries and cities, photos, and posts.

Differences with existing app

This app provides functionality similar to what we want to offer. However, this app offers the simple division like by N people only. However, the app we want to create will provide the Dutch payment feature in consideration of the difference spending per person. We also offer sharing function using Kakaotalk to share the results with your friends.

The features of the travel diaries and checklists are very similar to the apps we want to create. However, our apps provide SNS sharing so you can share these diaries with your friends. Also, existing apps have very uncomfortable UI. Our app will provide much improved UI as much as possible.

2-3. *Creditor (Dutch Pay App)*



This is one of the dutch pay application we can find on the Playstore easily. You can put the amount of money and need to select people who need to get paid back and gave money to someone else. The UI of this app looks interesting but actually it is not comfortable for the first-time users.

Differences with existing app

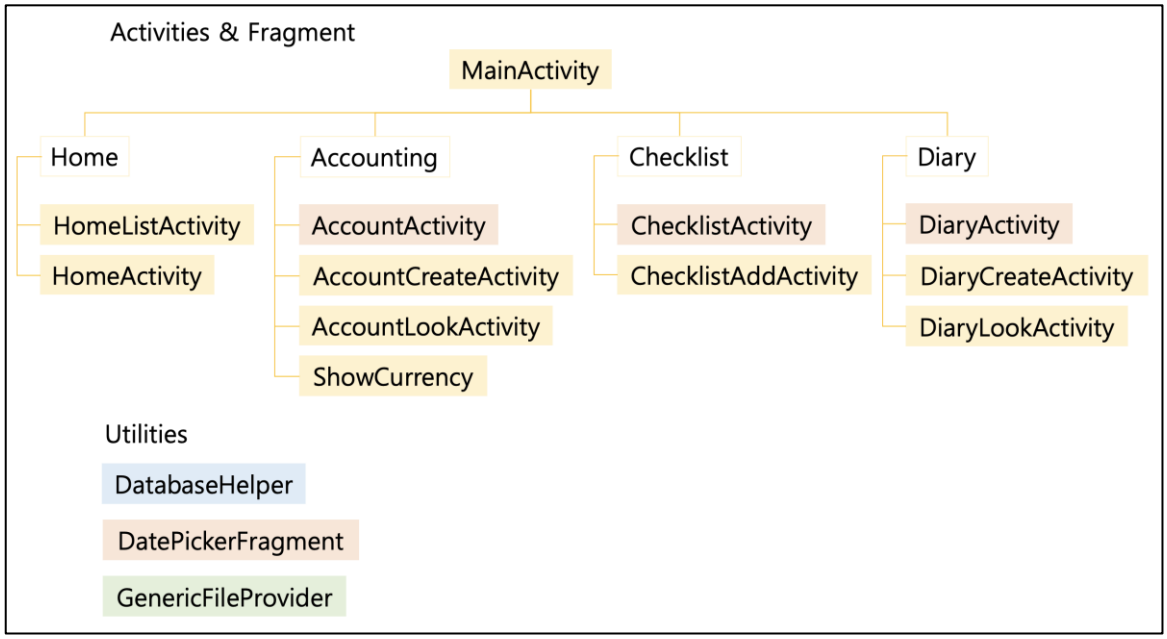
This app provides functionality similar to what we want to offer. However, this is not the travel application, so it does not consider the currency they paid. Also, the app cannot store the several list of the expenditure situation.

3. Implemented Functions

	Planned Functions	Completed Functions
Expenditure Record	O	O
Applying Exchange Rate	O	O
Calculating Accounting Book	O	O
Sharing Accounting Book	O	O
Checklist	O	O
Writing Diary	O	O
Editing & Deleting Diary	O	O
Sharing Diary	O	O
Administrating travels	O	O
Using Database	O	O
UI Design	O	O
Login	O	X
Sharing Travel with Friends	O	X

4. Code

** Class Diagram*



(Yellow: Activity / Red: Fragment / Blue: Database Helper / Green: Content Provider)

** Database*

travel		
_id	INTEGER	PRIMARY KEY
travel	TEXT	
members	TEXT	

checklist		
_id	INTEGER	PRIMARY KEY
title	TEXT	
travelname	TEXT	

accounting		
_id	INTEGER	PRIMARY KEY
date	TEXT	
title	TEXT	
participator	TEXT	
price	REAL	실수
currency	TEXT	
travelname	TEXT	

diary		
_id	INTEGER	PRIMARY KEY
date	TEXT	
title	TEXT	
picture	BLOB	byte[]
content	TEXT	
travelname	TEXT	

4-1. Calculating Accounting Book

4-1-1. Getting Currency

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);}
public void OnClick(View view) { new ExchangeRateTask().execute(); }

static class ExchangeRateTask extends AsyncTask<Void, Void, Void> {
    private String CLIENT_ID_EXCHANGE_RATE = "b9a8c1f1075e4fe883cfb9d2cc376d80";

    @Override
    public void onPostExecute(Void aVoid) { super.onPostExecute(aVoid); }

    @Override
    public Void doInBackground(Void... voids){
        String urlString = "https://openexchangerates.org/api/latest.json?app_id=" + CLIENT_ID_EXCHANGE_RATE;
        OkHttpClient client = new OkHttpClient();
        final Request request = new Request.Builder()
                .url(urlString)
                .build();
        client.newCall(request).enqueue(new Callback(){

            @Override
            public void onFailure(Call call, IOException e) { }

            @Override
            public void onResponse(Call call, Response response) throws IOException {
                AccountActivity.s = response.body().string();
                try{object = new JSONObject(AccountActivity.s);
                    object = object.getJSONObject("rates");
                }catch(JSONException e){
                    e.printStackTrace(); } } });
        return null;
    } }
```

4-1-2. Calculating Algorithm

```
private class putmap extends AsyncTask<Void, Void, Void> {
    @Override
    public void onPostExecute(Void aVoid) { super.onPostExecute(aVoid); }

    @Override
    public Void doInBackground(Void... voids){
        Map<String, Double> paid = new HashMap<~>();
        Map<String, Double> topay = new HashMap<~>();
        ArrayList<String> participators = new ArrayList<~>();
        String event = "";
        double eventpay = 0;
        Intent intent = new Intent(getActivity(), AccountLookActivity.class);
        constantsCursor = db.rawQuery( sql: "SELECT " + "*" +
                " FROM " + AccountingContract.ConstantEntry.TABLE_NAME +
                " WHERE " + AccountingContract.ConstantEntry.COLUMN_NAME_TRAVEL + " = " + "\"" +
                DataBaseHelper.now_travel + "\"",  selectionArgs: null);
        int count = constantsCursor.getCount();
        constantsCursor.moveToFirst();
        try{ while(true){
                if(s == null){
                    Thread.sleep( millis: 100);
                }else{
                    break; } }
```

```
object = new JSONObject(s);
object = object.getJSONObject("rates");
for(int i = 0; i < count; i++){
    String title = constantsCursor.getString( columnIndex: 2);
    String participator = constantsCursor.getString( columnIndex: 3);
    double price = constantsCursor.getDouble( columnIndex: 4);
    String currency = constantsCursor.getString( columnIndex: 5);

    //Fist paid
    if(paid.containsKey(participator)){
        double temp = paid.get(participator);
        double tem1 = price/object.getDouble(currency);
        double tem2 = tem1*object.getDouble( name: "KRW");
        temp += tem2;
        paid.remove(participator);
        paid.put(participator, temp);
    }else{
        double tem1 = price/object.getDouble(currency);
        double tem2 = tem1*object.getDouble( name: "KRW");
        paid.put(participator, tem2); }
```

```
//Second topay
if(event.equals(title)){
    eventpay += ((double)price)/object.getDouble(currency)*object.getDouble( name: "KRW");
    participators.add(participator);
}else{ for(int j = 0; j<participators.size(); j++){
        if(topay.containsKey(participators.get(j))){
            double temp = topay.get(participators.get(j));
            temp += eventpay/participators.size();
            topay.remove(participators.get(j));
            topay.put(participators.get(j), temp);
        }else{
            topay.put(participators.get(j), eventpay/participators.size()); } }
    eventpay = 0;
    participators.clear();
    event = title;
    eventpay = ((double)price)/object.getDouble(currency)*object.getDouble( name: "KRW");
    participators.add(participator); }
if(i == count-1){
    for(int j = 0; j<participators.size(); j++){
        if(topay.containsKey(participators.get(j))){
            double temp = topay.get(participators.get(j));
            temp += eventpay/participators.size();
            topay.remove(participators.get(j));
            topay.put(participators.get(j), temp);
        }else{
            topay.put(participators.get(j), eventpay/participators.size()); } } }
constantsCursor.moveToNext(); }
```

```
}catch(Exception e){ }
double[] divide = new double[listItemsac.size()];
int chc = 0;
for(String key : topay.keySet()){
    double value1 = topay.get(key);
    double value2 = paid.get(key);
    divide[chc] = value1 - value2;
    chc += 1; }
int[][] service = new int[listItemsac.size()][listItemsac.size()];
for(int i = 0; i<divide.length; i++){
    if(divide[i]<=0){
        continue;
    }else{ for(int j = 0; j<divide.length; j++){
            if(divide[j]>=0){
                continue;
            }else{
                if(divide[i]+divide[j]>0){
                    service[i][j] += (-divide[j]);
                    divide[i] += divide[j];
                    divide[j] = 0;
                }else{
                    service[i][j] += (divide[i]);
                    divide[j] += divide[i];
                    divide[i] = 0; } } } } }

for(int i = 0; i<listItemsac.size(); i++){
    for(int j = 0; j<listItemsac.size();j++){
        if(service[i][j] != 0){
            shout += paid.keySet().toArray()[i] + "님이 ";
            shout += paid.keySet().toArray()[j] + "님에게 " + service[i][j] + "원을, "; } } }
shout += "주시면 됩니다.";
s = null;
AccountActivity.checkResult = 1;
return null; } }
```

4-1-3. Sharing the result of calculation

```
public void shareKakaoTalk(){
    onRequestPermission();
    if (permessionCheck) {
        Intent kakao = new Intent(Intent.ACTION_SEND);
        kakao.setType("text/plain");
        try {
            kakao.putExtra(Intent.EXTRA_TEXT, shout);
            kakao.setPackage("com.kakao.talk");
            startActivity(kakao);
        } catch (ActivityNotFoundException e) {
            Toast.makeText(getContext(), text: "카카오톡이 설치되지 않았습니다.", Toast.LENGTH_SHORT).show();
        } catch (Exception e) {
            e.printStackTrace(); } } }
```

4-2. Sharing the diary using SNS

4-2-1. Sharing the diary to Instagram

```
public void shareInstagram() {
    onRequestPermission();
    if (permessionCheck) {
        long now = System.currentTimeMillis();
        Date date = new Date(now);
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat( pattern: "yyyyMMddHHmmss");
        String getTime = simpleDateFormat.format(date);
        ImageView picture = (ImageView) findViewById(R.id.diary_look_picture);
        Bitmap bitmap = ((BitmapDrawable)picture.getDrawable()).getBitmap();
        String storage = Environment.getExternalStorageDirectory().getAbsolutePath();
        String fileName = "nbangtravel"+getTime+".png";
        String folder = "/nBangTravel/";
        String fullPath = storage+folder;
        File file;
        try { file = new File(fullPath);
            if(!file.isDirectory()){
                file.mkdir(); }
            FileOutputStream fos = new FileOutputStream( name: fullPath+fileName);
            bitmap.compress(Bitmap.CompressFormat.PNG, quality: 100, fos);
            fos.flush();
            fos.close();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace(); }
        Intent share = new Intent(Intent.ACTION_SEND);
        share.setType("image/*");
        Uri uri = FileProvider.getUriForFile( context: this,
                authority: "soy.dow.nbang.nbangtravel.fileprovider",new File(fullPath, fileName));
        try {
            share.putExtra(Intent.EXTRA_STREAM, uri);
            share.setPackage("com.instagram.android");
            startActivity(share);
        } catch (ActivityNotFoundException e) {
            Toast.makeText( context: this, text: "인스타그램이 설치되지 않았습니다.", Toast.LENGTH_SHORT).show();
        } catch (Exception e) {
            e.printStackTrace(); } } }
```

4-2-2. Sharing the diary to Facebook

```
public void shareFacebook () {
    ImageView picture = (ImageView) findViewById(R.id.diary_look_picture);
    Bitmap bitmap = ((BitmapDrawable)picture.getDrawable()).getBitmap();
    CallbackManager callbackManager;
    ShareDialog shareDialog;
    callbackManager = CallbackManager.Factory.create();
    shareDialog = new ShareDialog( activity: this);
    shareDialog.registerCallback(callbackManager, new FacebookCallback<Sharer.Result>() {
        @Override
        public void onSuccess(Sharer.Result result) {
        }
        @Override
        public void onCancel() {
        }
        @Override
        public void onError(FacebookException error) {
        }
    });
    if (ShareDialog.canShow(SharePhotoContent.class)) {
        SharePhoto photo = new SharePhoto.Builder()
                .setBitmap(bitmap)
                .build();
        SharePhotoContent content = new SharePhotoContent.Builder()
                .addPhoto(photo)
                .build();
        shareDialog.show(content); } }
```

```
<provider android:authorities="com.facebook.app.FacebookContentProvider271734733529700"
    android:name="com.facebook.FacebookContentProvider"
    android:exported="true"/>

<meta-data android:name="com.facebook.sdk.ApplicationId"
    android:value="271734733529700"/>
```

4-3. Saving the diary (Storing the diary contents in the Database)

```
public void save() throws IOException {
    ImageView imageView = (ImageView)findViewById(R.id.diary_create_picture);
    Bitmap resized;
    byte[] barray = null;
    if(TextUtils.isEmpty(((TextView)findViewById(R.id.diary_create_date)).getText())){
        Toast.makeText( context: this,  text: "날짜를 반드시 입력해주세요", Toast.LENGTH_SHORT).show();
    }else if(TextUtils.isEmpty(((EditText)findViewById(R.id.diary_create_title)).getText())){
        Toast.makeText( context: this,  text: "다이어리의 제목을 반드시 입력해주세요", Toast.LENGTH_SHORT).show();
    }else if((BitmapDrawable)imageView.getDrawable() == null) {
        Toast.makeText( context: this,  text: "다이어리엔 사진이 필수!", Toast.LENGTH_SHORT).show();
    }else {
        resized = getResizedBitmap(((BitmapDrawable)imageView.getDrawable()).getBitmap(),
                imageView.getDrawable().getMinimumWidth());
        ByteArrayOutputStream stream = new ByteArrayOutputStream();
        resized.compress(Bitmap.CompressFormat.JPEG,  quality: 50, stream);
        barray = stream.toByteArray();
        resized.recycle();
        stream.close();
        String date = ((TextView)findViewById(R.id.diary_create_date)).getText().toString();
        String title =((EditText)findViewById(R.id.diary_create_title)).getText().toString();
        String content =  ((EditText)findViewById(R.id.diary_create_content)).getText().toString();
        String table_name = DataBaseHelper.now_travel;
        db = (new DataBaseHelper( context: this)).getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put(DiaryContract.ConstantEntry.COLUMN_NAME_DATE, date);
        values.put(DiaryContract.ConstantEntry.COLUMN_NAME_TITLE, title);
        values.put(DiaryContract.ConstantEntry.COLUMN_NAME_PICTURE, barray);
        values.put(DiaryContract.ConstantEntry.COLUMN_NAME_CONTENT, content);
        values.put(DiaryContract.ConstantEntry.COLUMN_NAME_TRAVEL, table_name);
        if(editDiary == 1) {
            db.update(DiaryContract.ConstantEntry.TABLE_NAME, values, whereClause: "_id = "+ EDIT_ID,  whereArgs: null);
            values.clear();
            Toast.makeText( context: this,  text: "수정되었습니다", Toast.LENGTH_SHORT).show();
            editDiary = 0;
        } else {
            db.insert(DiaryContract.ConstantEntry.TABLE_NAME,  nullColumnHack: null, values);
            values.clear();
            Toast.makeText( context: this,  text: "저장되었습니다", Toast.LENGTH_SHORT).show(); }
        MainActivity activ = (MainActivity)MainActivity.activ;
        activ.finish();
        Intent intent = new Intent( packageContext: this, MainActivity.class);
        MainActivity.check_ac = 88;
        startActivity(intent);
        finish(); } }
```

5. How to Use Application

This is the first page of the app that is showing existing travel history.

User can add new travel with travel name and participators.

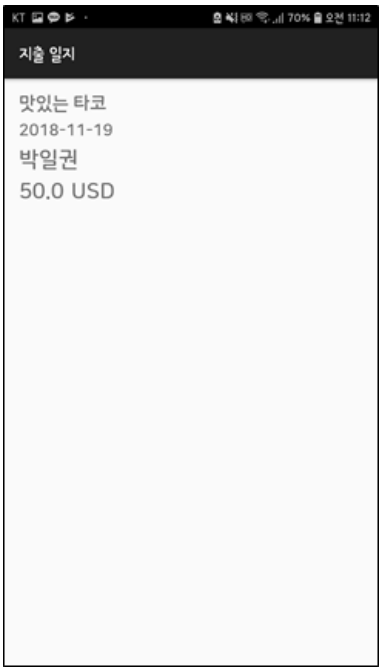
After creating a travel, user can check the new travel on the list.

User can add new expenditure record here

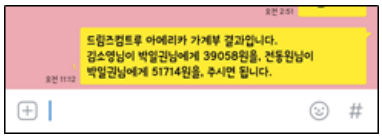
In this case, some participators paid the price.

In this case, only a few people participated in event.

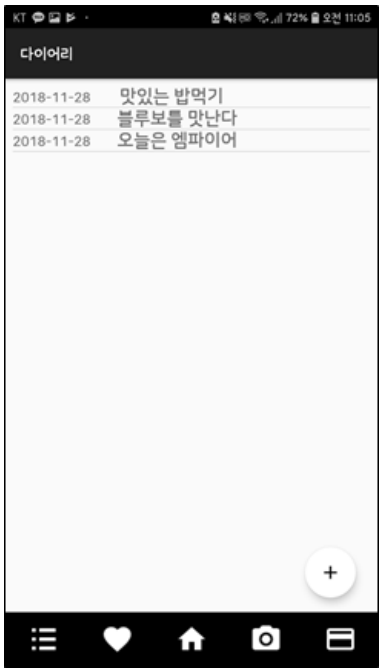
On the accounting book tab, user can check the expenditure list of each item.



Here user can see the detail of each expenditure.

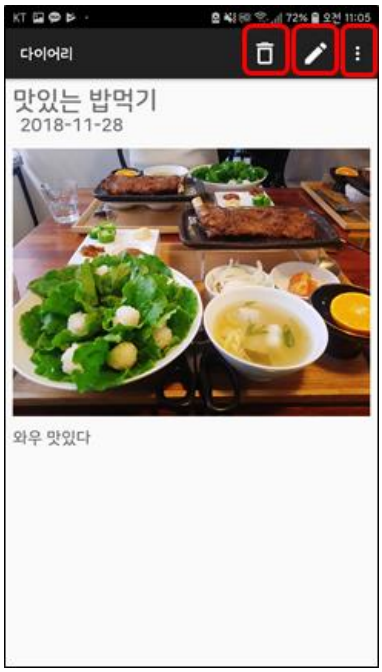


If user click the Kakaotalk button above,
the final result of the accounting book is calculated and sent into Kakaotalk.

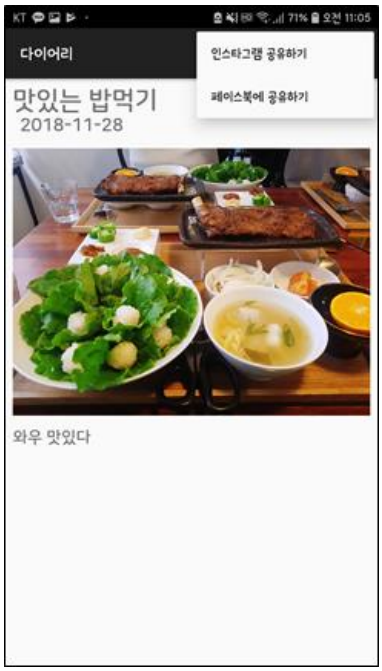


This is the Diary Tab.
User can see the list of diaries

User should input the title, date, picture, and documents.

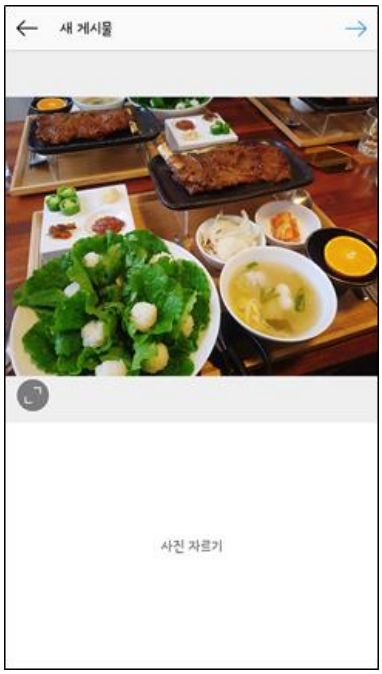


This is showing one of diaries already created.

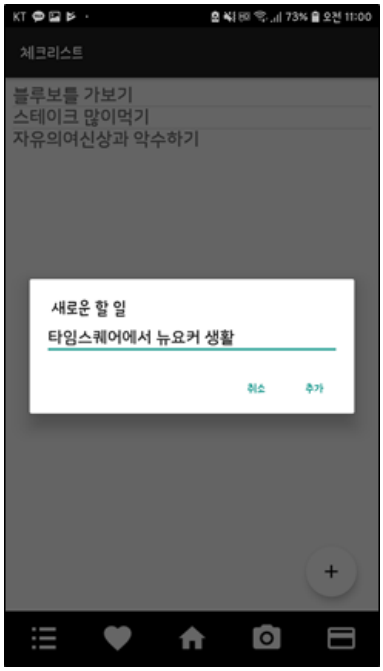


If user clicks the dot button on the right side,
it shows sharing button

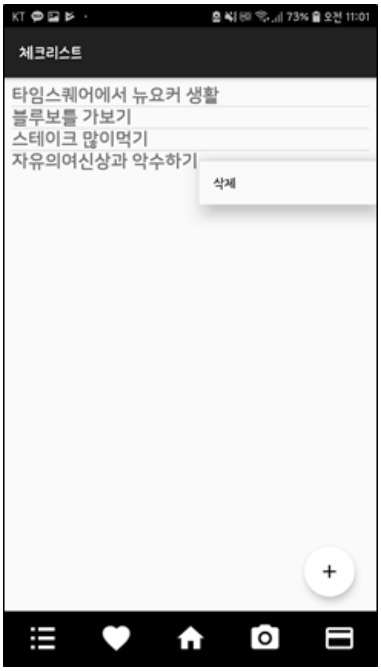
This is the snapshot of the sharing using Facebook.



This is the snapshot of the sharing using Instagram.



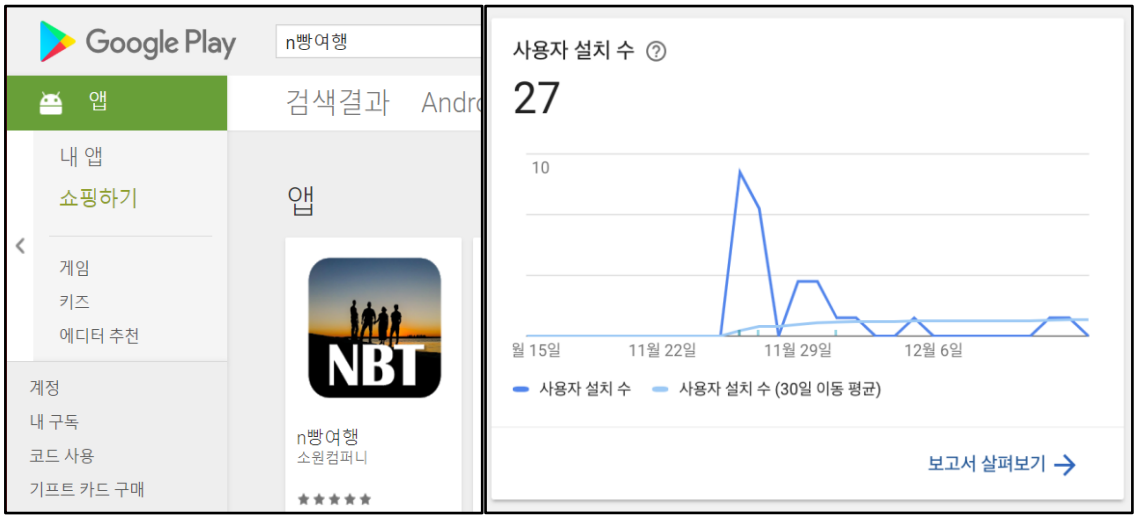
This is the Checklist Tab. User can add the important *TODO* list here.

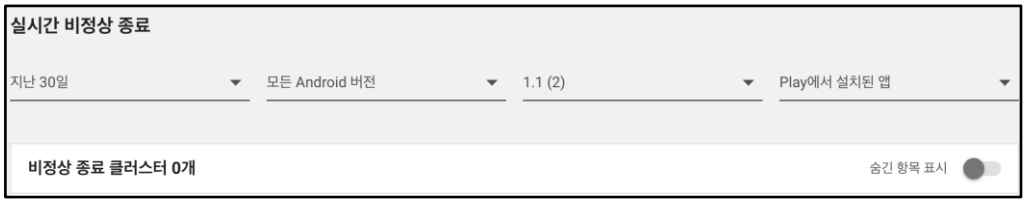


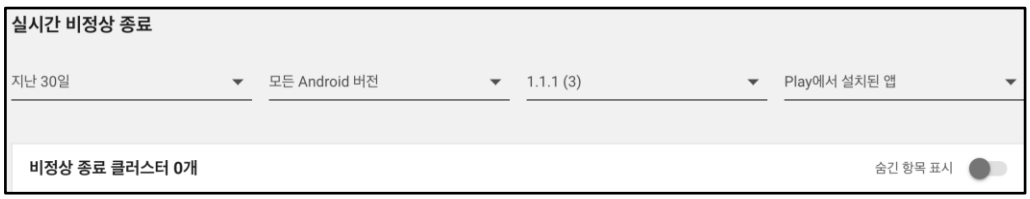
User can delete the checklist as well.

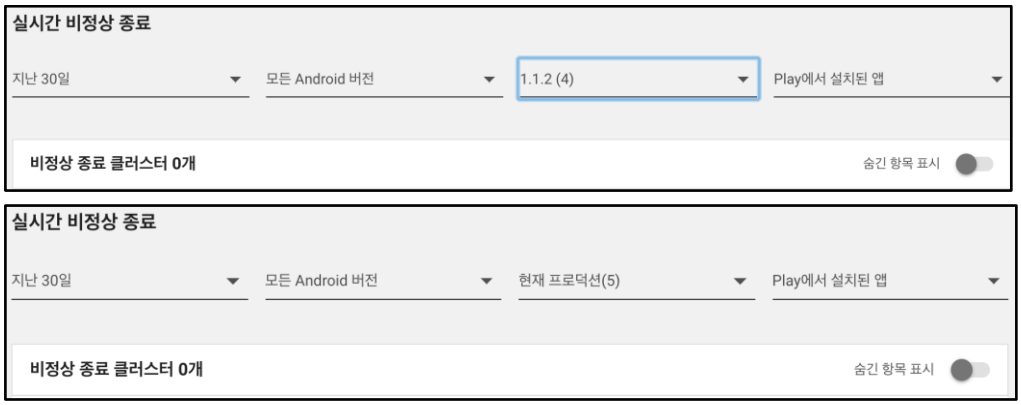
6. Release Result

Search ‘n 빵여행’ in Google Play Store and click download version 1.2. The app is available for API version 21(Lollipop) and above.









27 people downloaded the app, as of December 16. Actually, there were 2 abnormal terminations in first releasing version. However, from second releasing version in November 28 there were no abnormal terminations.

Github : https://github.com/nBangTravel/theTimeHasCome.git