

# REST API 연동

# ITEM LIST



# ITEM LIST

- ❖ 프로젝트 생성 – SpringBootAndroid
- ❖ AndroidManifest.xml 파일에 인터넷 권한 설정

```
<uses-permission android:name="android.permission.INTERNET"/>
```

```
<application  
    android:usesCleartextTraffic="true"
```

# ITEM LIST

- ❖ main\_activity.xml 파일을 수정

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    >
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="ITEM LIST"
        android:textSize="32sp"
        android:gravity="center"
        android:background="@color/teal_200"/>
    <ProgressBar
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/downloadview"/>
```

# ITEM LIST

- ❖ main\_activity.xml 파일을 수정

```
<ListView  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:id="@+id/listview"/>  
</LinearLayout>
```

# ITEM LIST

- ❖ 하나의 데이터를 표현하기 위한 Item 클래스 추가

```
public class Item implements Serializable {  
    private Long itemid;  
    private String itemname;  
    private Integer price;  
    private String description;  
    private String pictureurl;  
    private String email;
```

# ITEM LIST

- ❖ 하나의 데이터를 표현하기 위한 Item 클래스 추가

```
public Long getItemid() {  
    return itemid;  
}
```

```
public String getItemname() {  
    return itemname;  
}
```

```
public Integer getPrice() {  
    return price;  
}
```

```
public String getDescription() {  
    return description;  
}
```

```
public String getPictureurl() {  
    return pictureurl;  
}
```

```
public String getEmail() {  
    return email;  
}
```

# ITEM LIST

- ❖ 하나의 데이터를 표현하기 위한 Item 클래스 추가

```
public void setitemid(Long itemid) {  
    this.itemid = itemid;  
}
```

```
public void setItemname(String itemname) {  
    this.itemname = itemname;  
}
```

```
public void setPrice(Integer price) {  
    this.price = price;  
}
```

```
public void setDescription(String description) {  
    this.description = description;  
}
```

```
public void setPictureurl(String pictureurl) {  
    this.pictureurl = pictureurl;  
}
```

```
public void setEmail(String email) {  
    this.email = email;  
}
```

# ITEM LIST

- ❖ 하나의 데이터를 표현하기 위한 Item 클래스 추가

```
@Override  
public String toString() {  
    return "Item{" +  
        "itemid=" + itemid +  
        ", itemname=\"" + itemname + '\"' +  
        ", price=" + price +  
        ", description=\"" + description + '\"' +  
        ", pictureurl=\"" + pictureurl + '\"' +  
        ", email=\"" + email + '\"' +  
        '}';  
}  
}
```

# ITEM LIST

- ❖ 데이터를 로컬에 저장하기 위한 데이터베이스 연동 클래스를 추가 - ItemDB

```
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
```

```
class ItemDB extends SQLiteOpenHelper {
    public ItemDB(Context context) {
        super(context, "item.db", null, 1);
    }

    public void onCreate(SQLiteDatabase db) {
        db.execSQL("create table item(" +
            "itemid integer primary key, " +
            "itemname, " +
            "price integer, " +
            "description, " +
            "pictureurl, " +
            "email)");
    }

    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS item");
        onCreate(db);
    }
}
```

# ITEM LIST

- ❖ ListView에 출력할 하나의 셀 모양을 위한 item\_cell.xml 파일을 res/layout 디렉토리에 생성하고 작성

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="horizontal">
    <LinearLayout
        android:layout_width="0dp"
        android:layout_height="100dp"
        android:layout_weight="5"
        android:orientation="vertical">
        <TextView
            android:id="@+id/itemname"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_weight="1"
            android:text="아이템 이름"
            android:textSize="20dp" />
```

# ITEM LIST

- ❖ ListView에 출력할 하나의 셀 모양을 위한 item\_cell.xml 파일을 res/layout 디렉토리에 생성하고 작성

```
<TextView  
    android:id="@+id/price"  
    android:layout_width="wrap_content"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:text="아이템 가격"  
    android:textSize="20dp" />  
</LinearLayout>  
<ImageView  
    android:id="@+id/itemimage"  
    android:layout_width="0dp"  
    android:layout_height="match_parent"  
    android:layout_weight="5" />  
</LinearLayout>
```

# ITEM LIST

- ❖ ListView에 데이터 와 셀을 연결해주는 ItemAdapter 클래스를 생성하고 작성

```
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.Handler;
import android.os.Looper;
import android.os.Message;
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 java.io.InputStream;
import java.net.URL;
import java.util.List;
```

# ITEM LIST

- ❖ ListView에 데이터 와 셀을 연결해주는 ItemAdapter 클래스를 생성하고 작성

```
public class ItemAdapter extends BaseAdapter {  
    //뷰를 출력할 때 필요한 Context(문맥-어떤 작업을 하기 위해 필요한 정보를 저장한 객체) 변수  
    Context context;  
    //ListView에 출력할 데이터  
    List<Item> data;  
    //항목 뷰에 해당하는 레이아웃의 아이디를 저장할 변수  
    int layout;  
    //xml로 만들어진 레이아웃을 뷰로 변환하기 위한 클래스의 변수  
    LayoutInflater inflater;  
  
    public ItemAdapter(Context context, List<Item> data, int layout) {  
        super();  
        this.context = context;  
        this.data = data;  
        this.layout = layout;  
        inflater = (LayoutInflater)context.getSystemService(  
            Context.LAYOUT_INFLATER_SERVICE);  
    }  
}
```

# ITEM LIST

- ❖ ListView에 데이터 와 셀을 연결해주는 ItemAdapter 클래스를 생성하고 작성

```
@Override  
//출력할 데이터의 개수를 설정하는 메서드  
public int getCount() {  
    return data.size();  
}
```

```
@Override  
//항목 뷰에 보여질 문자열을 설정하는 메서드  
//position은 반복문이 수행될 때의 인덱스  
public Object getItem(int position) {  
    return data.get(position).getItemname();  
}
```

```
@Override  
//각 항목뷰의 아이디를 설정하는 메서드  
public long getItemId(int position) {  
    return position;  
}
```

# ITEM LIST

- ❖ ListView에 데이터 와 셀을 연결해주는 ItemAdapter 클래스를 생성하고 작성

```
@Override
```

```
//ListView에 출력될 실제 뷰의 모양을 설정하는 메서드
```

```
//convertView는 화면에 보여질 뷰인데 처음에는 null이 넘어오고 두번째 부터는  
//이전에 출력된 뷰가 넘어옵니다.
```

```
//인덱스마다 다른 뷰를 출력하고자 하면 convertView를 새로 만들지만
```

```
//모든 항목뷰의 모양이 같다면 처음 한번만 만들면 됩니다.
```

```
public View getView(int position, View convertView, ViewGroup parent) {
```

```
    final int pos = position;
```

```
    //convertView 생성
```

```
    if(convertView == null){
```

```
        //layout에 정의된 뷰를 parent에 넣을 수 있도록 View로 생성
```

```
        convertView = inflater.inflate(layout, parent, false);
```

```
}
```

```
//텍스트 출력
```

```
    TextView txtName = (TextView)convertView.findViewById(R.id.itemname);
```

```
    txtName.setText(data.get(pos).getItemname());
```

```
    TextView txtPrice = (TextView)convertView.findViewById(R.id.price);
```

```
    txtPrice.setText(data.get(pos).getPrice() + "원");
```

# ITEM LIST

- ❖ ListView에 데이터 와 셀을 연결해주는 ItemAdapter 클래스를 생성하고 작성

```
//이미지 출력
```

```
ImageView imageView = (ImageView)convertView.findViewById(R.id.itemimage);
```

```
Handler handler = new Handler(Looper.getMainLooper()){
```

```
    public void handleMessage(Message message){
```

```
        Bitmap bitmap = (Bitmap)message.obj;
```

```
        imageView.setImageBitmap(bitmap);
```

```
}
```

```
};
```

# ITEM LIST

- ❖ ListView에 데이터 와 셀을 연결해주는 ItemAdapter 클래스를 생성하고 작성

```
new Thread(){  
    public void run(){  
        try {  
            InputStream inputStream = new  
URL("http://172.30.30.93/member/download?path=" +  
                data.get(pos).getPictureurl()).openStream();  
            Bitmap bitmap = BitmapFactory.decodeStream(inputStream);  
            Message message = new Message();  
            message.obj = bitmap;  
            handler.sendMessage(message);  
        }catch(Exception e){}  
    }  
.start();  
return convertView;  
}  
}
```

# ITEM LIST

- ❖ MainActivity.java 파일에 인스턴스 변수 생성

```
private String updateTime;  
  
private int page;  
private int size;  
private int totalPage;  
  
private ItemDB itemDB;  
private List<Item> itemList;  
  
private ProgressBar downloadview;  
private ListView listView;  
private ItemAdapter itemAdapter;
```

# ITEM LIST

- ❖ MainActivity.java 파일에 ListView 출력을 위한 핸들러 생성

```
Handler handler = new Handler(Looper.getMainLooper()) {  
    public void handleMessage(Message msg) {  
        if (msg.what == 1) {  
            Snackbar.make(MainActivity.this.getWindow().getDecorView(), "데이터 업데이트",  
            Snackbar.LENGTH_LONG)  
                .show();  
            itemAdapter = new ItemAdapter(  
                MainActivity.this, itemList, R.layout.item_cell);  
            listView.setAdapter(itemAdapter);  
            downloadview.setVisibility(View.GONE);  
            try {  
                FileOutputStream fos = openFileOutput("updatetime.txt",  
                    Context.MODE_PRIVATE);  
                fos.write(updateTime.getBytes());  
                fos.close();  
            }catch(Exception e){  
                Log.e("업데이트 시간", "업데이트 시간을 기록할 수 없습니다.");  
            }  
        }  
    }  
};
```

# ITEM LIST

- ❖ MainActivity.java 파일에 데이터 관련 작업을 위한 스레드 생성

```
class DataDisplayThread extends Thread {  
    //다운로드 받은 문자열을 저장하기 위한 인스턴스 생성  
    StringBuilder sb = new StringBuilder();  
    @Override  
    public void run() {  
        try {  
            //다운로드 받을 주소 생성  
            URL url = new URL("http://172.30.30.93/item/updatedate");  
            //URL에 연결  
            HttpURLConnection con = (HttpURLConnection) url.openConnection();  
            //옵션 설정  
            con.setUseCaches(false);  
            con.setConnectTimeout(30000);
```

# ITEM LIST

- ❖ MainActivity.java 파일에 데이터 관련 작업을 위한 스레드 생성

```
//문자열을 다운로드 받기 위한 스트림을 생성
```

```
BufferedReader br = new BufferedReader(new  
InputStreamReader(con.getInputStream()));
```

```
//문자열을 읽어서 저장
```

```
while (true) {
```

```
    String line = br.readLine();
```

```
    if (line == null)
```

```
        break;
```

```
    sb.append(line + "\n");
```

```
}
```

```
//읽은 데이터 확인
```

```
Log.e("updatetime", sb.toString());
```

```
//사용한 스트림과 연결 해제
```

```
br.close();
```

```
con.disconnect();
```

```
JSONObject object = new JSONObject(sb.toString());
```

```
updateTime = object.getString("updatedate");
```

```
Log.e("업데이트 한 시간", updateTime);
```

# ITEM LIST

- ❖ MainActivity.java 파일에 데이터 관련 작업을 위한 스레드 생성

```
String localUpdateTime = null;  
try {  
    FileInputStream fis = openFileInput("updatetime.txt");  
    byte[] data = new byte[fis.available()];  
    fis.read(data);  
    fis.close();  
    localUpdateTime = new String(data);  
} catch (FileNotFoundException e) {  
    Log.e("업데이트 한 시간", "파일이 없습니다.");  
}
```

# ITEM LIST

- ❖ MainActivity.java 파일에 데이터 관련 작업을 위한 스레드 생성

```
if(updateTime.equals(localUpdateTime)){
    Log.e("서버 와의 비교", "시간이 동일하므로 업데이트 할 필요가 없습니다.");
    FileInputStream fis = openFileInput("totalpage.txt");
    byte[] data = new byte[fis.available()];
    fis.read(data);
    totalPage = Integer.parseInt(new String(data));
    Log.e("전체 페이지 개수_1", totalPage + "");
    fis.close();

    fos = openFileOutput("page.txt",
        Context.MODE_PRIVATE);
    fos.write(("" + page).getBytes());
    fos.close();
}else{
    Log.e("서버 와의 비교", "시간이 다르므로 업데이트를 해야 할 필요가 있습니다.");
}

//다운로드 받을 주소 생성

url = new URL("http://172.30.30.93/item/list?page=" + page + "&size=" + size);
//URL에 연결
con = (HttpURLConnection) url.openConnection();
//옵션 설정
con.setUseCaches(false);
con.setConnectTimeout(30000);
```

# ITEM LIST

- ❖ MainActivity.java 파일에 데이터 관련 작업을 위한 스레드 생성

```
//문자열을 다운로드 받기 위한 스트림을 생성  
br = new BufferedReader(new InputStreamReader(con.getInputStream()));  
sb = new StringBuilder();  
//문자열을 읽어서 저장  
while (true) {  
    String line = br.readLine();  
    if (line == null)  
        break;  
    sb.append(line + "\n");  
}  
//읽은 데이터 확인  
Log.e("listdata", sb.toString());  
//사용한 스트림과 연결 해제  
br.close();  
con.disconnect();
```

# ITEM LIST

- ❖ MainActivity.java 파일에 데이터 관련 작업을 위한 스레드 생성

```
JSONObject data = new JSONObject(sb.toString());
String error = data.getString("error");
if(error.equals("null")){
    Log.e("에러 메시지", "에러 메시지가 없음");

    totalPage = data.getInt("totalPage");
    FileOutputStream fos = openFileOutput("totalpage.txt",
        Context.MODE_PRIVATE);
    fos.write(("" + totalPage).getBytes());
    fos.close();
```

# ITEM LIST

- ❖ MainActivity.java 파일에 데이터 관련 작업을 위한 스레드 생성

```
JSONArray ar = data.getJSONArray("itemList");

SQLiteDatabase db = itemDB.getWritableDatabase();
db.delete("item", null, null);
for(int i=0; i<ar.length(); i++){
    JSONObject itemObj = ar.getJSONObject(i);

    ContentValues row = new ContentValues();
    row.put("itemid", itemObj.getLong("itemid"));
    row.put("itemname", itemObj.getString("itemname"));
    row.put("price", itemObj.getInt("price"));
    row.put("description", itemObj.getString("description"));
    row.put("pictureurl", itemObj.getString("pictureurl"));
    row.put("email", itemObj.getString("email"));

    db.insert("item", null, row);
}
```

# ITEM LIST

- ❖ MainActivity.java 파일에 데이터 관련 작업을 위한 스레드 생성

```
    }else{
        Log.e("에러", error);
    }
}
```

# ITEM LIST

- ❖ MainActivity.java 파일에 데이터 관련 작업을 위한 스레드 생성

```
SQLiteDatabase db = itemDB.getReadableDatabase();
Cursor cursor= db.rawQuery(
    "select itemid, itemname, price, description, pictureurl, email " +
    "from item order by itemid desc", null);

itemList.clear();
while (cursor.moveToNext()){
    Item item = new Item();
    item.setItemid(cursor.getLong(0));
    item.setItemname(cursor.getString(1));
    item.setPrice(cursor.getInt(2));
    item.setDescription(cursor.getString(3));
    item.setPictureurl(cursor.getString(4));
    item.setEmail(cursor.getString(5));
    itemList.add(item);
}
```

# ITEM LIST

- ❖ MainActivity.java 파일에 데이터 관련 작업을 위한 스레드 생성

```
Message message = new Message();
message.what = 1;
handler.sendMessage(message);
} catch (Exception e) {
    Log.e("다운로드 실패", e.getMessage());
}
}
```

# ITEM LIST

- ❖ MainActivity.java 파일의 onCreate 메서드에서 초기화 작업 수행

```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
  
    try {  
        FileInputStream fis = openFileInput("page.txt");  
        byte[] data = new byte[fis.available()];  
        fis.read(data);  
        fis.close();  
        page = Integer.parseInt(new String(data));  
    }catch (Exception e){  
        page = 1;  
    }      size = 15;  
  
    itemDB = new ItemDB(this);  
    itemList = new ArrayList<>();
```

# ITEM LIST

- ❖ MainActivity.java 파일의 onCreate 메서드에서 초기화 작업 수행

```
listView = (ListView)findViewById(R.id.listview);
downloadview = (ProgressBar)findViewById(R.id.downloadview);

//데이터를 ListView에 출력할 수 있도록 Adapter에 주입
itemAdapter = new ItemAdapter(
    this, itemList, R.layout.item_cell);
listView.setAdapter(itemAdapter);

new DataDisplayThread().start();

}
```

# Scroll

- ❖ MainActivity.java 파일에 스크롤을 처리하기 위한 속성을 추가

```
//가장 하단에서 스크롤 했는지 확인하기 위한 프로퍼티
```

```
private Boolean lastItemVisibleFlag = false;
```

# Scroll

- ❖ MainActivity.java 파일의 onCreate 메서드에서 ListView 의 스크롤 이벤트 처리 코드 추가

```
listView.setOnScrollListener(new AbsListView.OnScrollListener() {  
    @Override  
    public void onScrollStateChanged(AbsListView view, int scrollState) {  
        if (scrollState == AbsListView.OnScrollListener.SCROLL_STATE_IDLE &&  
lastItemVisibleFlag) {  
            if (page >= totalPage) {  
                Toast.makeText(MainActivity.this, "더이상의 데이터가 없습니다.",  
Toast.LENGTH_LONG).show();  
        }  
    }  
}
```

# Scroll

- ❖ MainActivity.java 파일의 onCreate 메서드에서 ListView 의 스크롤 이벤트 처리 코드 추가

```
    } else {
        page = page + 1;
        downloadview.setVisibility(View.VISIBLE);
        new Thread() {
            public void run() {
                try {
                    //다운로드 받을 주소 생성

                    URL url = new URL("http://172.30.30.93/item/list?page=" + page +
"&size=" + size);
                    //URL에 연결
                    HttpURLConnection con = (HttpURLConnection)
url.openConnection();
                    //옵션 설정
                    con.setUseCaches(false);
                    con.setConnectTimeout(30000);
```

# Scroll

- ❖ MainActivity.java 파일의 onCreate 메서드에서 ListView 의 스크롤 이벤트 처리 코드 추가

```
//문자열을 다운로드 받기 위한 스트림을 생성  
BufferedReader br = new BufferedReader(new  
InputStreamReader(con.getInputStream()));  
StringBuilder sb = new StringBuilder();  
//문자열을 읽어서 저장  
while (true) {  
    String line = br.readLine();  
    if (line == null)  
        break;  
    sb.append(line + "\n");  
}  
//사용한 스트림과 연결 해제  
br.close();  
con.disconnect();
```

# Scroll

- ❖ MainActivity.java 파일의 onCreate 메서드에서 ListView 의 스크롤 이벤트 처리 코드 추가

```
JSONObject data = new JSONObject(sb.toString());
String error = data.getString("error");
if (error.equals("null")) {
    JSONArray ar = data.getJSONArray("itemList");
    SQLiteDatabase db = itemDB.getWritableDatabase();
    db.delete("item", null, null);
    for (int i = 0; i < ar.length(); i++) {
        JSONObject itemObj = ar.getJSONObject(i);

        ContentValues row = new ContentValues();
        row.put("itemid", itemObj.getLong("itemid"));
        row.put("itemname", itemObj.getString("itemname"));
        row.put("price", itemObj.getInt("price"));
        row.put("description", itemObj.getString("description"));
        row.put("pictureurl", itemObj.getString("pictureurl"));
        row.put("email", itemObj.getString("email"));

        db.insert("item", null, row);
    }
} else {
    Log.e("에러", error);
}
```

# Scroll

- ❖ MainActivity.java 파일의 onCreate 메서드에서 ListView 의 스크롤 이벤트 처리 코드 추가

```
SQLiteDatabase db = itemDB.getReadableDatabase();
Cursor cursor = db.rawQuery(
    "select itemid, itemname, price, description, pictureurl, email "
+
    "from item order by itemid desc", null);
while (cursor.moveToNext()) {
    Item item = new Item();
    item.setItemid(cursor.getLong(0));
    item.setItemname(cursor.getString(1));
    item.setPrice(cursor.getInt(2));
    item.setDescription(cursor.getString(3));
    item.setPictureurl(cursor.getString(4));
    item.setEmail(cursor.getString(5));
    itemList.add(item);
}
```

# Scroll

- ❖ MainActivity.java 파일의 onCreate 메서드에서 ListView 의 스크롤 이벤트 처리 코드 추가

```
Message message = new Message();
message.what = 1;
handler.sendMessage(message);

} catch (Exception e) {
}
}.start();
}
}
}

@Override
public void onScroll(AbsListView view, int firstVisibleItem, int visibleItemCount, int
totalCount) {
    lastItemVisibleFlag = totalCount > 0 && firstVisibleItem + visibleItemCount >=
totalCount;
}
});
```

# Pull To Refresh

- ❖ build.gradle 파일에 의존성 추가 : 자주 변경되므로 검색해서 추가  
implementation 'androidx.swiperefreshlayout:swiperefreshlayout:1.1.0'

# Pull To Refresh

- ❖ activity\_main.xml 파일의 ListView 수정

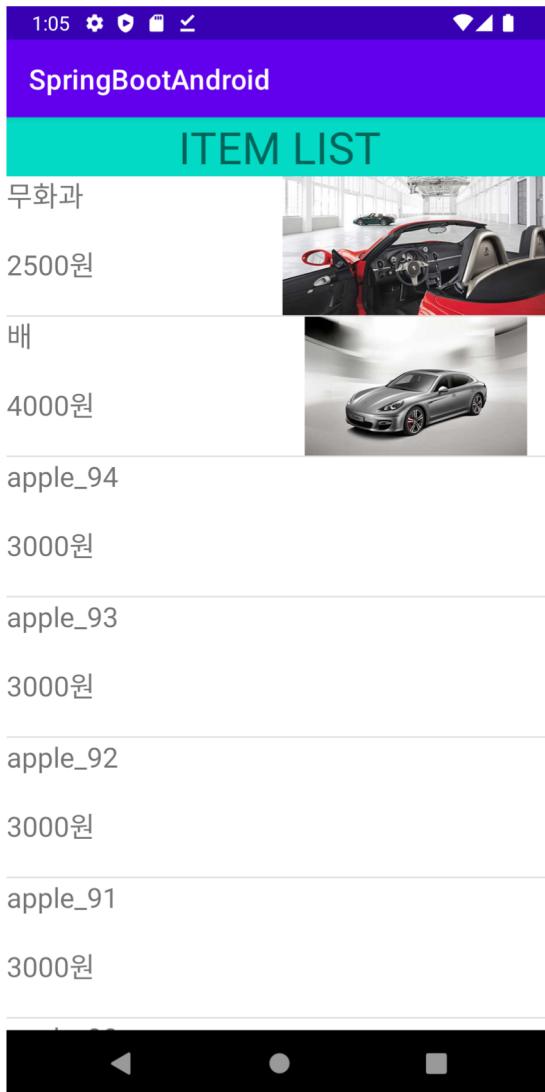
```
<androidx.swiperefreshlayout.widget.SwipeRefreshLayout  
    android:id="@+id/swipe_layout"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
    <ListView  
        android:id="@+id/listview"  
        android:layout_width="match_parent"  
        android:layout_height="match_parent" />  
</androidx.swiperefreshlayout.widget.SwipeRefreshLayout>
```

# Pull To Refresh

- ❖ activity\_main.xml 파일의 ListView 수정

```
SwipeRefreshLayout swipeRefreshLayout = (SwipeRefreshLayout)
findViewById(R.id.swipe_layout);
swipeRefreshLayout.setOnRefreshListener(new SwipeRefreshLayout.OnRefreshListener() {
    @Override
    public void onRefresh() {
        downloadview.setVisibility(View.GONE);
        swipeRefreshLayout.setRefreshing(false);
        page = 1;
        new DataDisplayThread().start();
    }
});
```

# 상세보기



# 상세보기

- ❖ 상세 보기를 위한 Activity를 추가 - DetailActivity

# 상세보기

- ❖ MainActivity 클래스의 onCreate 메서드에 ListView의 셀을 선택했을 때 호출되는 코드를 작성

```
listView.setOnItemClickListener(new AdapterView.OnItemClickListener() {  
    @Override  
    //첫번째 매개변수는 이벤트가 발생한 ListView  
    //두번째 매개변수는 이벤트가 발생한 항목 뷰  
    //세번째 매개변수는 이벤트가 발생한 인덱스  
    //네번째 매개변수는 이벤트가 발생한 항목 뷰의 아이디  
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {  
        Item item = itemList.get(position);  
        Intent intent = new Intent(MainActivity.this, DetailActivity.class);  
        intent.putExtra("item", item);  
        startActivity(intent);  
    }  
});
```

# 상세보기

- ❖ activity\_detail.xml 파일을 수정

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <Button
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="뒤로"
        android:textSize="32dp"
        android:gravity="center_horizontal"
        android:id="@+id/back"/>
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="32dp"
        android:gravity="center_horizontal"
        android:id="@+id/itemname"/>
```

# 상세보기

- ❖ activity\_detail.xml 파일을 수정

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content">  
    <TextView  
        android:layout_width="0dp"  
        android:layout_height="wrap_content"  
        android:layout_weight="2"  
        android:textSize="24dp"  
        android:text="가격"/>  
    <TextView  
        android:layout_width="0dp"  
        android:layout_height="wrap_content"  
        android:layout_weight="3"  
        android:textSize="24dp"  
        android:id="@+id/price"/>  
</LinearLayout>
```

# 상세보기

- ❖ activity\_detail.xml 파일을 수정

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content">  
    <TextView  
        android:layout_width="0dp"  
        android:layout_height="wrap_content"  
        android:layout_weight="2"  
        android:textSize="24dp"  
        android:text="설명"/>  
    <TextView  
        android:layout_width="0dp"  
        android:layout_height="wrap_content"  
        android:layout_weight="3"  
        android:textSize="24dp"  
        android:id="@+id/description"/>  
    </LinearLayout>  
    <ImageView  
        android:layout_width="match_parent"  
        android:layout_height="match_parent"  
        android:id="@+id/picture"/>  
</LinearLayout>
```

# 상세보기

- ❖ DetailActivity.java 파일에 속성 과 이미지를 출력할 핸들러 생성

```
private String pictureurl;  
private ImageView imageView;
```

```
Handler imageHandler = new Handler(Looper.getMainLooper()){  
    public void handleMessage(Message message){  
        Bitmap bit = (Bitmap)message.obj;  
        imageView.setImageBitmap(bit);  
    }  
};
```

# 상세보기

- ❖ DetailActivity.java 파일에 이미지를 다운로드 받고 출력하기 위한 스레드 생성

```
class ImageThread extends Thread{  
    public void run(){  
        try{  
            InputStream inuptStream = new URL("http://172.30.30.93/member/download?path=" + pictureurl).openStream();  
            Bitmap bit = BitmapFactory.decodeStream(inuptStream);  
            inuptStream.close();  
            Message message = new Message();  
            message.obj = bit;  
            imageHandler.sendMessage(message);  
        }catch(Exception e){  
            Log.e("이미지 가져오기 에러", e.getLocalizedMessage());  
        }  
    }  
}
```

# 상세보기

- ❖ DetailActivity.java 파일의 onCreate 메서드 수정

```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_detail);  
  
    Intent intent = getIntent();  
    Item item = (Item)intent.getSerializableExtra("item");  
  
    TextView itemname = (TextView)findViewById(R.id.itemname);  
    TextView price = (TextView)findViewById(R.id.price);  
    TextView description = (TextView)findViewById(R.id.description);  
    ImageView imageView = (ImageView)findViewById(R.id.picture);  
  
    itemname.setText(item.getItemname());  
    price.setText(item.getPrice() + "원");  
    description.setText(item.getDescription());  
    pictureurl = item.getPictureurl();
```

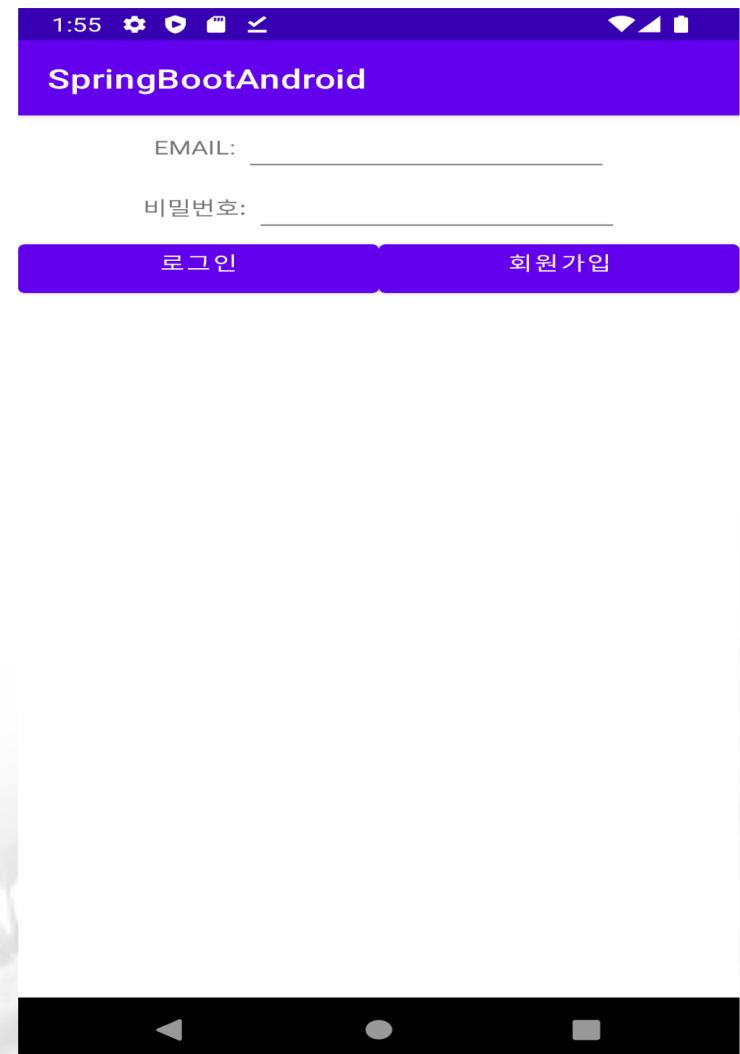
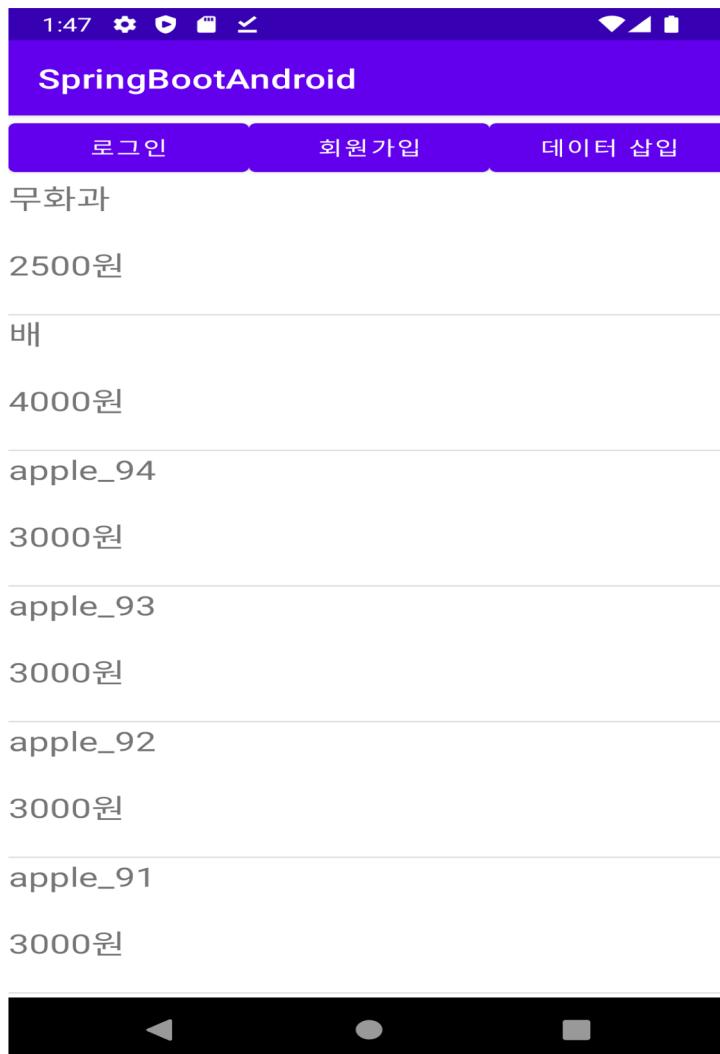
# 상세보기

- ❖ DetailActivity.java 파일의 onCreate 메서드 수정

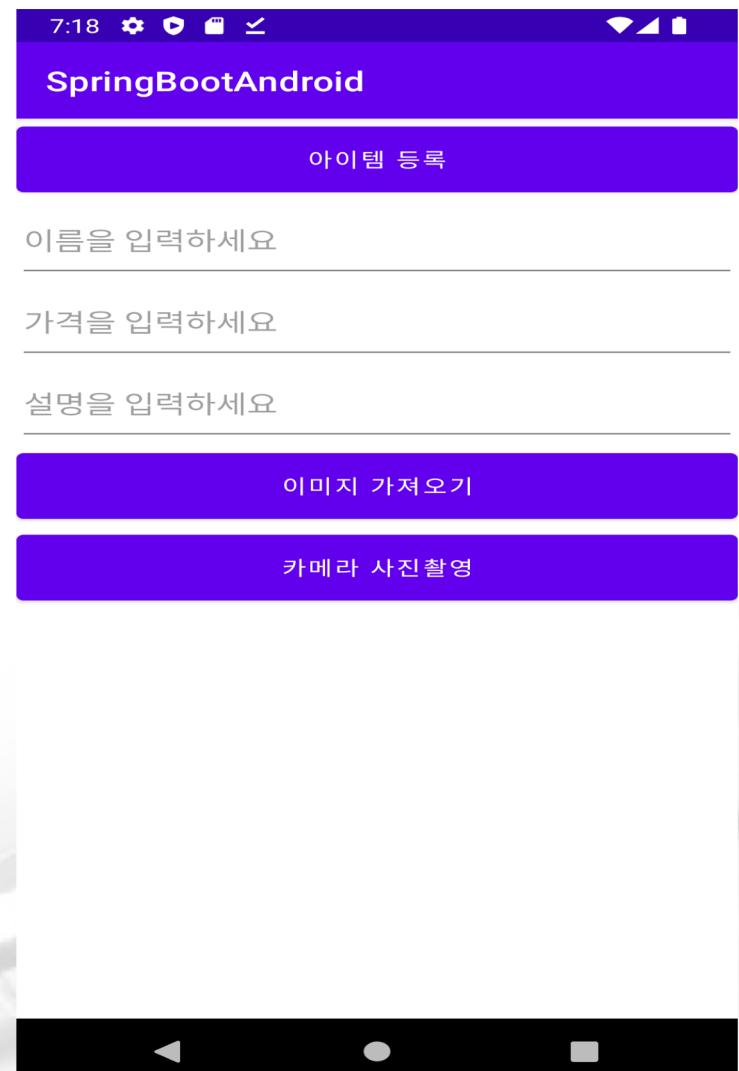
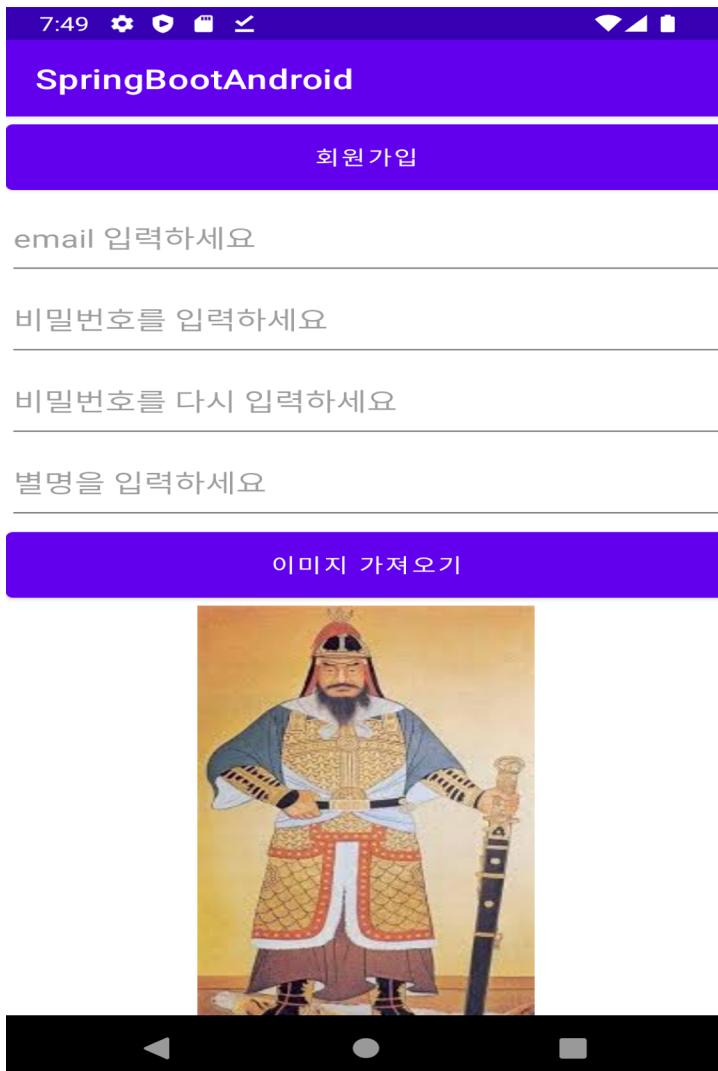
```
new ImageThread().start();
Button backBtn = (Button)findViewById(R.id.back);
backBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        finish();
    }
});
```

}

# 화면 이동



# 화면 이동



# 화면 이동

- ❖ 실행 가능한 Activity 생성 – LoginActivity, MemberRegisterActivity, ItemRegisterActivity

# 화면 이동

- ❖ activity\_login.xml 파일에 디자인 수정

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".LoginActivity"
    android:orientation="vertical">
```

# 화면 이동

- ❖ activity\_login.xml 파일에 디자인 수정

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:gravity="center_horizontal"  
    android:orientation="horizontal" >  
    <TextView  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:text=" EMAIL: " />  
    <EditText  
        android:id="@+id/emailinput"  
        android:layout_width="200dp"  
        android:layout_height="wrap_content"  
        android:inputType="textEmailAddress"  
        android:text="" />  
</LinearLayout>
```

# 화면 이동

- ❖ activity\_login.xml 파일에 디자인 수정

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:gravity="center_horizontal"  
    android:orientation="horizontal" >  
    <TextView  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:text="비밀번호: " />  
    <EditText  
        android:id="@+id/pwinput"  
        android:inputType="textPassword"  
        android:layout_width="200dp"  
        android:layout_height="wrap_content"  
        android:text="" />  
</LinearLayout>
```

# 화면 이동

- ❖ activity\_login.xml 파일에 디자인 수정

```
<LinearLayout  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:gravity="center_horizontal"  
    android:orientation="horizontal" >  
    <Button  
        android:id="@+id	btnlogin"  
        android:layout_width="0dp"  
        android:layout_height="wrap_content"  
        android:layout_weight="1"  
        android:gravity="center_horizontal"  
        android:text="로그인"/>  
    <Button  
        android:id="@+id	btnlogout"  
        android:layout_width="0dp"  
        android:layout_height="wrap_content"  
        android:layout_weight="1"  
        android:gravity="center_horizontal"  
        android:text="회원가입"/>  
</LinearLayout>  
</LinearLayout>
```

# 화면 이동

- ❖ activity\_member\_register.xml 파일에 디자인 수정

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MemberRegisterActivity"
    android:orientation="vertical">
    <Button
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:text="회원가입"
        android:id="@+id/registerbtn"/>
```

# 화면 이동

- ❖ activity\_member\_register.xml 파일에 디자인 수정

```
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:hint="email 입력하세요"  
    android:id="@+id/emailinput"/>  
  
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:inputType="textPassword"  
    android:hint="비밀번호를 입력하세요"  
    android:id="@+id/pwinput"  
    />  
  
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:inputType="textPassword"  
    android:hint="비밀번호를 다시 입력하세요"  
    android:id="@+id/pwconfirminput"  
    />
```

# 화면 이동

- ❖ activity\_member\_register.xml 파일에 디자인 수정

```
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:hint="별명을 입력하세요"  
    android:id="@+id/nicknameinput"/>  
  
<Button  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:text="이미지 가져오기"  
    android:id="@+id/imagebtn"/>  
  
<ImageView  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="5"  
    android:id="@+id/image"/>  
  
</LinearLayout>
```

# 화면 이동

- ❖ activity\_item\_register.xml 파일에 디자인 수정

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ItemRegisterActivity"
    android:orientation="vertical">
    <Button
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:text="아이템 등록"
        android:id="@+id/registerbtn"/>
    <EditText
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:hint="이름을 입력하세요"
        android:id="@+id/itemnameinput"/>
```

# 화면 이동

- ❖ activity\_item\_register.xml 파일에 디자인 수정

```
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:inputType="numberDecimal"  
    android:hint="가격을 입력하세요"  
    android:id="@+id/priceinput"  
/>  
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:hint="설명을 입력하세요"  
    android:id="@+id/descriptioninput"/>  
<Button  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:text="이미지 가져오기"  
    android:id="@+id/imagebtn"/>
```

# 화면 이동

- ❖ activity\_item\_register.xml 파일에 디자인 수정

```
<Button  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:text="카메라 사진촬영"  
    android:id="@+id/camerabtn"/>  
  
<ImageView  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="5"  
    android:id="@+id/image"/>  
  
</LinearLayout>
```

# 화면 이동

- ❖ activity\_item\_register.xml 파일에 디자인 수정

```
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:inputType="numberDecimal"  
    android:hint="가격을 입력하세요"  
    android:id="@+id/priceinput"  
/>  
<EditText  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:hint="설명을 입력하세요"  
    android:id="@+id/descriptioninput"/>  
<Button  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="1"  
    android:text="이미지 가져오기"  
    android:id="@+id/imagebtn"/>
```

# 화면 이동

- ❖ activity\_item\_register.xml 파일에 디자인 수정

```
<ImageView  
    android:layout_width="match_parent"  
    android:layout_height="0dp"  
    android:layout_weight="5"  
    android:id="@+id/image"/>  
  
</LinearLayout>
```

# MEMBER

- ❖ activity\_main.xml 파일을 수정

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    >
```

# 화면 이동

- ❖ activity\_main.xml 파일을 수정

```
<androidx.appcompat.widget.LinearLayoutCompat  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content">  
    <Button  
        android:layout_width="0dp"  
        android:layout_height="match_parent"  
        android:layout_weight="1"  
        android:text="로그인"  
        android:id="@+id/loginbtn"/>  
    <Button  
        android:layout_width="0dp"  
        android:layout_height="match_parent"  
        android:layout_weight="1"  
        android:text="회원가입"  
        android:id="@+id/memberregisterbtn"/>  
    <Button  
        android:layout_width="0dp"  
        android:layout_height="match_parent"  
        android:layout_weight="1"  
        android:text="데이터 삽입"  
        android:id="@+id/itemregisterbtn"/>  
</androidx.appcompat.widget.LinearLayoutCompat>
```

# 화면 이동

- ❖ activity\_main.xml 파일을 수정

```
<ProgressBar  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:id="@+id/downloadview"/>  
<androidx.swiperefreshlayout.widget.SwipeRefreshLayout  
    android:id="@+id/swipe_layout"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
  
<ListView  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:id="@+id/listview"/>  
</androidx.swiperefreshlayout.widget.SwipeRefreshLayout>  
</LinearLayout>
```

# 화면 이동

- ❖ MainActivity.java 파일에 인스턴스 변수 선언

```
private Button loginBtn, memberRegisterBtn, itemRegisterBtn;
```

# 화면 이동

- ❖ MainActivity.java 파일의 onCreate 메서드에 추가

```
loginBtn = (Button)findViewById(R.id.loginbtn);
memberRegisterBtn = (Button)findViewById(R.id.memberregisterbtn);
itemRegisterBtn = (Button)findViewById(R.id.itemregisterbtn);
```

# 화면 이동

- ❖ MainActivity.java 파일의 onCreate 메서드에 추가

```
try {  
    FileInputStream fis = openFileInput("login.txt");  
    byte[] data = new byte[fis.available()];  
    while (fis.read(data) != -1) {}  
    fis.close();  
  
    String content = new String(data);  
    String [] ar = content.split(":");  
    loginBtn.setText("로그아웃");  
    memberRegisterBtn.setText("회원정보수정");  
} catch (Exception e) {}
```

# 화면 이동

- ❖ MainActivity.java 파일의 onCreate 메서드에 추가

```
loginBtn.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View v) {
        if(loginBtn.getText().equals("로그인")){
            Intent intent = new Intent(MainActivity.this, LoginActivity.class);
            startActivity(intent);
        }else{
            deleteFile("login.txt");
            loginBtn.setText("로그인");
            memberRegisterBtn.setText("회원가입");
        }
    }
});
```

# 화면 이동

- ❖ MainActivity.java 파일의 onCreate 메서드에 추가

```
memberRegisterBtn.setOnClickListener(new View.OnClickListener(){  
    @Override  
    public void onClick(View v) {  
        Intent intent = new Intent(MainActivity.this, MemberRegisterActivity.class);  
        intent.putExtra("task",memberRegisterBtn.getText());  
        startActivity(intent);  
    }  
});
```

# 화면 이동

- ❖ MainActivity.java 파일의 onCreate 메서드에 추가

```
itemRegisterBtn.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View v) {
        if(loginBtn.getText().equals("로그아웃")) {
            Intent intent = new Intent(MainActivity.this, ItemRegisterActivity.class);
            startActivity(intent);
        }else{
            Toast.makeText(MainActivity.this, "로그인을 해야 합니다.",
Toast.LENGTH_LONG).show();
            Intent intent = new Intent(MainActivity.this, LoginActivity.class);
            startActivity(intent);
        }
    }
});
```

# 로그인

- ❖ LoginActivity.java 파일에 인스턴스 변수 선언

```
EditText emailinput, pwInput;  
Button loginBtn, memberRegisterBtn;
```

```
String email, nickname, imageurl;
```

# 로그인

- ❖ LoginActivity.java 파일에 핸들러 작성

```
//메시지 출력을 위한 핸들러  
Handler handler = new Handler(Looper.getMainLooper()){  
    @Override  
    public void handleMessage(Message message){  
        Toast.makeText(LoginActivity.this, "로그인 실패", Toast.LENGTH_SHORT).show();  
    }  
};
```

# 로그인

- ❖ LoginActivity.java 파일에 로그인 처리를 위한 스레드를 추가

```
//로그인 처리를 위한 스레드
class LoginThread extends Thread{
    String json;
    @Override
    public void run(){
        try{
            //다운로드 받을 주소 생성
            URL url = new URL("http://192.168.0.6/member/login");
            //URL에 연결
            HttpURLConnection con = (HttpURLConnection) url.openConnection();
            con.setRequestMethod("POST");
            con.setUseCaches(false);

            //파라미터 생성
            String data = URLEncoder.encode("email", "UTF-8") + "=" +
URLEncoder.encode(emailinput.getText().toString().trim(), "UTF-8");
            data += "&" + URLEncoder.encode("password", "UTF-8") + "=" +
URLEncoder.encode(pwInput.getText().toString().trim(), "UTF-8");
            OutputStreamWriter wr = new OutputStreamWriter(con.getOutputStream());
            wr.write(data);
            wr.flush();
```

# 로그인

- ❖ LoginActivity.java 파일에 로그인 처리를 위한 스레드를 추가

```
//문자열을 다운로드 받기 위한 스트림을 생성  
BufferedReader br =  
    new BufferedReader(  
        new InputStreamReader(  
            con.getInputStream()));  
StringBuilder sb = new StringBuilder();  
//문자열을 읽어서 저장  
while (true) {  
    String line = br.readLine();  
    if (line == null)  
        break;  
    sb.append(line + "\n");  
}  
//사용한 스트림과 연결 해제  
br.close();  
con.disconnect();  
json = sb.toString();  
}catch(Exception e){  
    Log.e("로그인 예외", e.getMessage());  
    Message message = new Message();  
    message.obj = "로그인 처리 에러";  
    handler.sendMessage(message);  
}
```

# 로그인

- ❖ LoginActivity.java 파일에 로그인 처리를 위한 스레드를 추가

```
if(json != null){  
    try{  
        JSONObject object = new JSONObject(json);  
        String error = object.getString("error");  
        if(error.equals("null")){  
            email = object.getString("email");  
            nickname = object.getString("name");  
            imageurl = object.getString("imageurl");  
  
            FileOutputStream fos = openFileOutput("login.txt",  
                Context.MODE_PRIVATE);  
            String str = email + ":" + nickname + ":" + imageurl;  
            fos.write(str.getBytes());  
            fos.close();  
            Intent intent = new Intent(LoginActivity.this, MainActivity.class);  
            startActivity(intent);  
        }  
        else{  
            Log.e("로그인", "로그인 실패");  
            handler.sendMessage(0);  
        }  
    }  
}
```

# 로그인

- ❖ LoginActivity.java 파일에 로그인 처리를 위한 스레드를 추가

```
        }catch(Exception e){  
            Log.e("로그인", "로그인 실패");  
            handler.sendMessage(0);  
        }  
    }  
}
```

# 로그인

- ❖ LoginActivity.java 파일의 onCreate 메서드에 처리를 위한 코드를 추가

```
emailinput = (EditText)findViewById(R.id.emailinput);
pwInput = (EditText)findViewById(R.id.pwinput);

loginBtn = (Button)findViewById(R.id.loginbtn);
loginBtn.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View v) {
        new LoginThread().start();
    }
});

memberRegisterBtn = (Button)findViewById(R.id.memberregisterbtn);
memberRegisterBtn.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View v) {
        Intent intent = new Intent(LoginActivity.this, MemberRegisterActivity.class);
        intent.putExtra("task",memberRegisterBtn.getText());
        startActivity(intent);
    }
});
```

# 회원가입

- ❖ 삽입할 이미지에 해당하는 파일을 res 디렉토리에 raw 디렉토리를 생성하고 복사 – musa.jpeg

# 회원가입

- ❖ MemberRegisterActivity.java 파일에 뷰를 위한 인스턴스 변수 작성

```
private EditText emailinput, pwinput, pwconfirminput, nicknameinput;  
private Button imagebtn, registerbtn;  
private ImageView image;
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일의 onCreate 메서드에 뷰를 찾아오는 코드를 작성

```
emailinput = (EditText)findViewById(R.id.emailinput);
pwinput = (EditText)findViewById(R.id.pwinput);
pwconfirminput = (EditText)findViewById(R.id.pwconfirminput);
nicknameinput = (EditText)findViewById(R.id.nicknameinput);

imagebtn = (Button)findViewById(R.id.imagebtn);
registerbtn = (Button)findViewById(R.id.registerbtn);

image = (ImageView)findViewById(R.id.image);
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일에 핸들러 작성

```
//회원가입 여부를 출력하는 핸들러
Handler handler = new Handler(Looper.getMainLooper()){
    @Override
    public void handleMessage(Message message){
        String result = (String)message.obj;
        switch(message.what){
            case 0:
                Toast.makeText(MemberRegisterActivity.this, result,
Toast.LENGTH_SHORT).show();
                break;
            case 1:
                Toast.makeText(MemberRegisterActivity.this, result,
Toast.LENGTH_SHORT).show();
        }
    }
}
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일에 핸들러 작성

```
//키보드 관리 객체 가져오기  
InputMethodManager imm = (InputMethodManager) getSystemService(  
    INPUT_METHOD_SERVICE);  
  
imm.hideSoftInputFromWindow(emailinput.getWindowToken(),0);  
imm.hideSoftInputFromWindow(pwinput.getWindowToken(),0);  
imm.hideSoftInputFromWindow(pwconfirminput.getWindowToken(),0);  
imm.hideSoftInputFromWindow(nicknameinput.getWindowToken(),0);  
  
emailinput.setText("");  
pwinput.setText("");  
pwconfirminput.setText("");  
nicknameinput.setText("");  
  
Intent intent = new Intent(MemberRegisterActivity.this, MainActivity.class);  
startActivity(intent);  
  
break;  
}  
};  
};
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일에 회원 가입을 위한 스레드 클래스를 생성

```
//회원가입 요청을 전송하는 스레드
class ThreadEx extends Thread{
    String json;
    @Override
    public void run(){
        try{
            //다운로드 받을 주소 생성
            URL url = new URL("http://172.30.76.70/member/register");
            //URL에 연결

            HttpURLConnection con = (HttpURLConnection) url.openConnection();

            //파일을 제외한 파라미터 만들기
            String[] data = {emailinput.getText().toString(), pwinput.getText().toString(),
                nicknameinput.getText().toString()};
            String[] dataName = {"email", "password", "name"};
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일에 회원 가입을 위한 스레드 클래스를 생성

```
// boundary생성, 여기서는 고정값이지만 되도록이면 실행할때마다 다른값을 할당
String lineEnd = "\r\n";
String boundary = UUID.randomUUID().toString();
//연결 옵션 설정
con.setRequestMethod("POST");
con.setReadTimeout(10000);
con.setConnectTimeout(10000);
con.setDoOutput(true);
con.setDoInput(true);
con.setUseCaches(false);

//파일 첨부가 있는 경우 생성
con.setRequestProperty("ENCTYPE", "multipart/form-data");
con.setRequestProperty("Content-Type","multipart/form-
data;boundary=" + boundary);

String delimiter = "--" + boundary + lineEnd; // --androidupload\r\n
StringBuffer postDataBuilder = new StringBuffer();
//파라미터를 하나로 만들기
for(int i=0;i<data.length;i++){
    postDataBuilder.append(delimiter);
    postDataBuilder.append("Content-Disposition: form-data; name=\"" +
dataName[i] + "\""+lineEnd+lineEnd+data[i]+lineEnd);
}
}
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일에 회원가입을 위한 스레드 클래스를 생성

```
// 파일 파라미터 생성  
String fileName = nicknameinput.getText() + ".png";  
if(image.getDrawable() != null){  
    postDataBuilder.append(delimiter);  
    postDataBuilder.append("Content-Disposition: form-data; name=\""+  
"image\";filename=\""+fileName +"\""+ lineEnd);  
}
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일에 회원 가입을 위한 스레드 클래스를 생성

```
//파라미터 전송
DataOutputStream ds = new DataOutputStream(con.getOutputStream());
ds.write(postDataBuilder.toString().getBytes());
if(image.getDrawable() != null){
    //파일 업로드
    ds.writeBytes(lineEnd);
    InputStream fres = getResources().openRawResource(R.raw.musa);
    byte[] buffer = new byte[fres.available()];
    int length = -1;
    while((length=fres.read(buffer)) != -1){
        ds.write(buffer,0,length);
    }
    ds.writeBytes(lineEnd);
    ds.writeBytes(lineEnd);
    ds.writeBytes("--" + boundary + "--" + lineEnd); // requestbody end
    fres.close();
}
else {
    ds.writeBytes(lineEnd);
    ds.writeBytes("--" + boundary + "--" + lineEnd); // requestbody end
}
ds.flush();
ds.close();
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일에 회원 가입을 위한 스레드 클래스를 생성

```
//문자열을 다운로드 받기 위한 스트림을 생성  
BufferedReader br = new BufferedReader(new  
InputStreamReader(con.getInputStream()));  
StringBuilder sb = new StringBuilder();  
//문자열을 읽어서 저장  
while (true) {  
    String line = br.readLine();  
    if (line == null)  
        break;  
    sb.append(line + "\n");  
}  
//사용한 스트림과 연결 해제  
br.close();  
con.disconnect();  
json = sb.toString();  
}catch(Exception e){  
    Log.e("회원가입 예외", e.getMessage());  
    Message message = new Message();  
    message.obj = "회원가입 에러";  
    message.what = 0;  
    handler.sendMessage(message);  
}
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일에 회원 가입을 위한 스레드 클래스를 생성

```
if(json != null){  
    try{  
        Log.e("넘어온 데이터", json);  
        JSONObject object = new JSONObject(json);  
        Log.e("데이터",object.toString());  
        String error = object.getString("error");  
        Message message = new Message();  
        if(error.equals("null")) {  
            message.obj = "회원가입 성공";  
            message.what = 1;  
        }else{  
            message.obj = error;  
            message.what = 0;  
        }  
        handler.sendMessage(message);  
    }  
}
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일에 회원가입을 위한 스레드 클래스를 생성

```
    catch(Exception e){  
        Log.e("파싱 예외", e.getMessage());  
        Message message = new Message();  
        message.obj = "파싱 에러";  
        message.what = 0;  
        handler.sendMessage(message);  
    }  
}  
}  
}
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일의 onCreate 메서드에 버튼 클릭 이벤트 처리 코드를 작성

```
imagebtn.setOnClickListener(new View.OnClickListener(){  
    @Override  
    public void onClick(View v) {  
        Bitmap bitmap = BitmapFactory.decodeResource(getResources(), R.raw.musa);  
        image.setImageBitmap(bitmap);  
    }  
});
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일의 onCreate 메서드에 버튼 클릭 이벤트 처리 코드를 작성

```
registerbtn.setOnClickListener(new View.OnClickListener(){  
    @Override  
    public void onClick(View v) {  
        Message errorMessage = new Message();  
  
        String email = emailinput.getText().toString().trim();  
        String pw = pwinput.getText().toString().trim();  
        String pwconfirm = pwconfirminput.getText().toString().trim();  
        String nickname = nicknameinput.getText().toString().trim();  
    }  
});
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일의 onCreate 메서드에 버튼 클릭 이벤트 처리 코드를 작성

```
if(email.length() == 0){  
    errorMessage.obj = "email은 비어있을 수 없습니다.>";  
    handler.sendMessage(errorMessage);  
    return;  
}  
else{  
    String regex = "^[_a-z0-9-]+(.[_a-z0-9-]+)*@(?>WWW+WW.)+WWW+$";  
    Pattern p = Pattern.compile(regex);  
    Matcher m = p.matcher(email);  
    if(m.matches() == false) {  
        errorMessage.obj = "email 형식과 일치하지 않습니다.";  
        errorMessage.what = 0;  
        handler.sendMessage(errorMessage);  
        return;  
    }  
}
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일의 onCreate 메서드에 버튼 클릭 이벤트 처리 코드를 작성

```
if(pw.length() == 0){  
    errorMessage.obj = "비밀번호는 비어있을 수 없습니다.>";  
    errorMessage.what = 0;  
    handler.sendMessage(errorMessage);  
    return;  
}  
else{  
    String regex = "^(?=.*[a-z])(?=.*[A-Z])(?=.*[\\W\\d])(?=.*[$@#!%*?&])[A-Za-  
z\\W\\d$#!%*?&]{8,}";  
    Pattern p = Pattern.compile(regex);  
    Matcher m = p.matcher(pw);  
    if(m.matches() == false) {  
        errorMessage.obj = "비밀번호는 영문 대소문자 1개 이상 특수문자 1개 숫자  
1개 이상으로 만들어져야 합니다. ";  
        errorMessage.what = 0;  
        handler.sendMessage(errorMessage);  
        return;  
    }  
}
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일의 onCreate 메서드에 버튼 클릭 이벤트 처리 코드를 작성

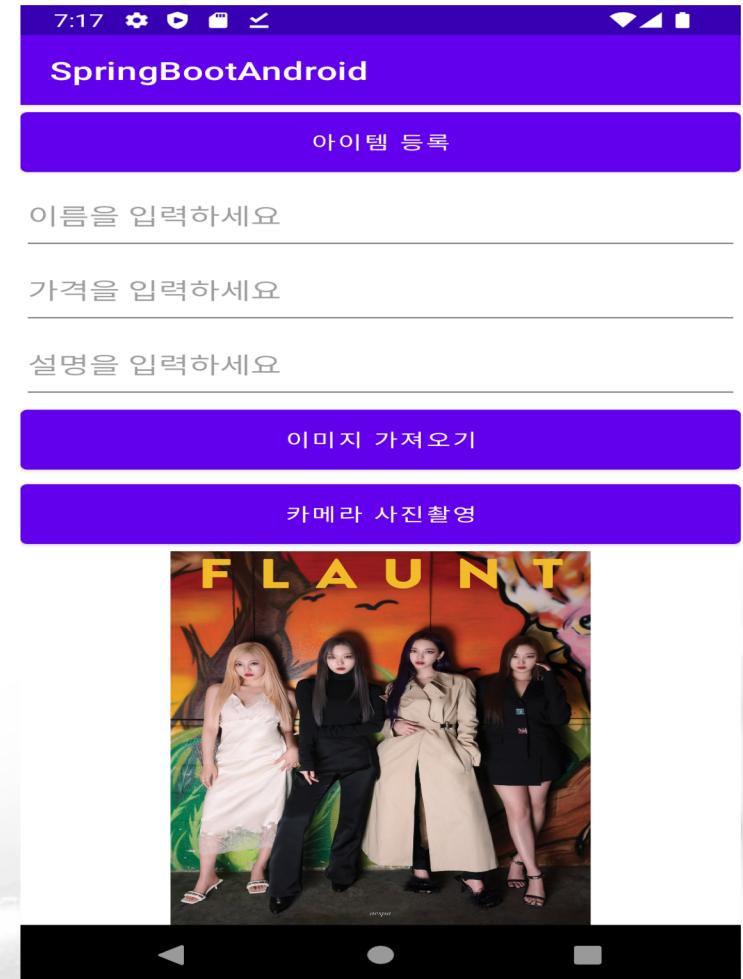
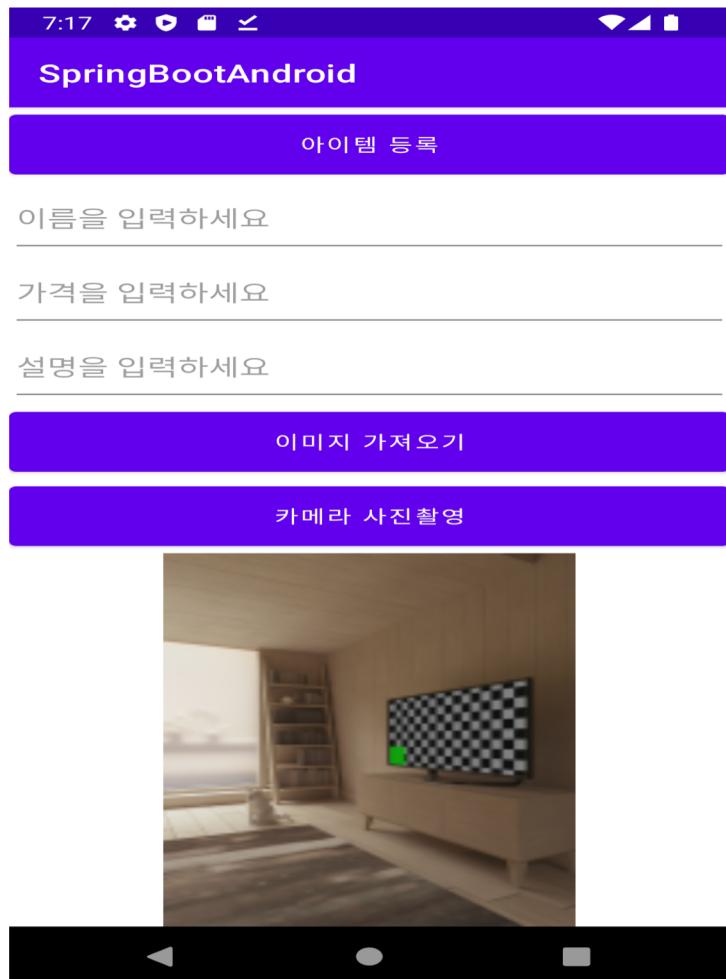
```
if(pw.equals(pwconfirm) == false) {  
    errorMessage.obj = "2개의 비밀번호가 다릅니다. ";  
    errorMessage.what = 0;  
    handler.sendMessage(errorMessage);  
    return;  
}
```

# 회원가입

- ❖ MemberRegisterActivity.java 파일의 onCreate 메서드에 버튼 클릭 이벤트 처리 코드를 작성

```
if(nickname.length() < 2){  
    errorMessage.obj = "별명은 2자 이상이어야 합니다.";  
    errorMessage.what = 0;  
    handler.sendMessage(errorMessage);  
    return;  
}  
else{  
    String regex = "[0-9]|[a-z]|[A-Z]|[가-힣]";  
    for(int i = 0; i < nickname.length(); i++) {  
        String ch = nickname.charAt(i) + "";  
        Pattern p = Pattern.compile(regex);  
        Matcher m = p.matcher(ch);  
        if (m.matches() == false) {  
            errorMessage.obj = "별명은 영문 숫자 한글만 사용해야 합니다.";  
            errorMessage.what = 0;  
            handler.sendMessage(errorMessage);  
            return;  
        }  
    }  
}  
  
new ThreadEx().start();  
}  
});
```

# ITEM 삽입



# ITEM 삽입

- ❖ ItemRegisterActivity.java 파일에 인스턴스 변수 선언

```
//화면에 디자인 한 뷰의 참조를 저장하기 위한 뷰  
Button registerbtn, imagebtn, camerabtn;  
EditText itemnameinput, priceinput, descriptioninput;  
ImageView image;
```

# ITEM 삽입

- ❖ ItemRegisterActivity.java 파일의 onCreate 메서드에 뷰를 찾아오는 코드를 추가

```
registerbtn = (Button)findViewById(R.id.registerbtn);
imagebtn = (Button) findViewById(R.id.imagebtn);
camerabtn = (Button)findViewById(R.id.camerabtn);

itemnameinput = (EditText)findViewById(R.id.itemnameinput);
priceinput = (EditText)findViewById(R.id.priceinput);
descriptioninput = (EditText)findViewById(R.id.descriptioninput);

image = (ImageView)findViewById(R.id.image);
```

# ITEM 삽입

- ❖ ItemRegisterActivity.java 파일에 gallery 와 camera 에서 사진을 선택할 수 있는 Launcher 인스턴스를 생성

```
ActivityResultLauncher<Intent> imageLauncher = registerForActivityResult(new  
ActivityResultContracts.StartActivityForResult(), new ActivityResultCallback<ActivityResult>() {  
    @Override  
    public void onActivityResult(ActivityResult result) {  
        if (result.getResultCode() == RESULT_OK) {  
            Intent intent = result.getData();  
            Uri uri = intent.getData();  
            image.setImageURI(uri);  
        }  
    }  
});
```

# ITEM 삽입

- ❖ ItemRegisterActivity.java 파일에 gallery 와 camera 에서 사진을 선택할 수 있는 Launcher 인스턴스를 생성

```
ActivityResultLauncher<Intent> cameraLauncher = registerForActivityResult(new  
ActivityResultContracts.StartActivityForResult(), new ActivityResultCallback<ActivityResult>() {  
    @Override  
    public void onActivityResult(ActivityResult result) {  
        if (result.getResultCode() == RESULT_OK) {  
            Intent intent = result.getData();  
            Bundle extras = intent.getExtras();  
            Bitmap imageBitmap = (Bitmap) extras.get("data");  
            image.setImageBitmap(imageBitmap);  
        }  
    }  
});
```

# ITEM 삽입

- ❖ ItemRegisterActivity.java 파일의 onCreate 메서드에 이미지 와 카메라 버튼의 클릭 이벤트 처리 코드 작성

```
imagebtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Intent intent = new Intent();
        intent.setType("image/*");
        intent.setAction(Intent.ACTION_GET_CONTENT);
        imageLauncher.launch(intent);
    }
});

camerabtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        Log.e("클릭", "클릭");
        Intent imageTakeIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
        cameraLauncher.launch(imageTakeIntent);
    }
});
```

# ITEM

- ❖ ItemRegisterActivity.java 파일에 메시지 출력을 위한 Handler 구현

```
//데이터 삽입 결과를 출력하기 위한 핸들러
Handler handler = new Handler(Looper.getMainLooper()){
    @Override
    public void handleMessage(Message msg){
        //유효성 검사 메시지를 출력
        if(msg.what == 0){
            String result = (String)msg.obj;
            Toast.makeText(ItemRegisterActivity.this, result, Toast.LENGTH_SHORT).show();
        }
        //삽입 결과를 출력
        else if(msg.what == 1) {
            String error = (String)msg.obj;
            if (error.equals("null")) {
                Toast.makeText(ItemRegisterActivity.this, "삽입 성공",
Toast.LENGTH_SHORT).show();
                itemnameinput.setText("");
                priceinput.setText("");
                descriptioninput.setText("");
            }
        }
    }
}
```

# ITEM

- ❖ ItemRegisterActivity.java 파일에 메시지 출력을 위한 Handler 구현

```
//키보드 관리 객체 가져오기
InputMethodManager imm = (InputMethodManager) getSystemService(
    INPUT_METHOD_SERVICE);
imm.hideSoftInputFromWindow(itemnameinput.getWindowToken(), 0);
imm.hideSoftInputFromWindow(priceinput.getWindowToken(), 0);
imm.hideSoftInputFromWindow(descriptioninput.getWindowToken(), 0);

Intent intent = new Intent(ItemRegisterActivity.this, MainActivity.class);
startActivity(intent);
} else {
    Toast.makeText(ItemRegisterActivity.this, error, Toast.LENGTH_SHORT).show();
}
}
};

};
```

# ITEM

- ❖ ItemRegisterActivity.java 파일에 서버에 데이터를 전송하는 스레드 코드 구현

```
class ThreadEx extends Thread{
    String json;

    @Override
    public void run(){
        Message message = new Message();
        try{
            FileInputStream fis = openFileInput("login.txt");
            byte[] b = new byte[fis.available()];
            fis.read(b);
            fis.close();
            String member = new String(b);
            String email = member.split(":")[0];

            //다운로드 받을 주소 생성
            URL url = new URL("http://172.30.76.70/item/register");
            //URL에 연결

            HttpURLConnection con = (HttpURLConnection) url.openConnection();
```

# ITEM

- ❖ ItemRegisterActivity.java 파일에 서버에 데이터를 전송하는 스레드 코드 구현

```
//파일을 제외한 파라미터 만들기  
String[] data = {itemnameinput.getText().toString(),  
    priceinput.getText().toString(),  
    descriptioninput.getText().toString(),  
    email};  
String[] dataName = {"itemname", "price", "description", "email"};  
  
// boundary생성 실행할때마다 다른값을 할당 : 파일 업로드가 있을 때는 반드시  
생성  
String lineEnd = "\r\n";  
String boundary = UUID.randomUUID().toString();  
  
// 연결 객체 옵션 설정  
con.setRequestMethod("POST");  
con.setConnectTimeout(10000);  
con.setUseCaches(false);
```

# ITEM

- ❖ ItemRegisterActivity.java 파일에 서버에 데이터를 전송하는 스레드 코드 구현

```
// 파일 업로드가 있는 경우 설정  
con.setRequestProperty("ENCTYPE", "multipart/form-data");  
con.setRequestProperty("Content-Type","multipart/form-  
data;boundary=" + boundary);  
//파라미터 생성  
String delimiter = "--" + boundary + lineEnd; // --androidupload₩r₩n  
StringBuffer postDataBuilder = new StringBuffer();  
for(int i=0;i<data.length;i++){  
    postDataBuilder.append(delimiter);  
    postDataBuilder.append("Content-Disposition: form-data; name=\"" +  
dataName[i] + "\""+lineEnd+lineEnd+data[i]+lineEnd);  
}  
  
String fileName = itemnameinput.getText().toString() + ".png"; //이미지가  
존재하는 경우  
//String fileName = null; //이미지가 존재하지 않는 경  
// 파일이 존재할 때에만 생성  
if(image.getDrawable() != null){  
    postDataBuilder.append(delimiter);  
    postDataBuilder.append("Content-Disposition: form-data; name=\"" +  
"image" + "\";filename=\"" + fileName +"\""+ lineEnd);  
}
```

# ITEM

- ❖ ItemRegisterActivity.java 파일에 서버에 데이터를 전송하는 스레드 코드 구현

```
//파라미터 전송  
DataOutputStream ds = new DataOutputStream(con.getOutputStream());  
ds.write(postDataBuilder.toString().getBytes());  
  
if(image.getDrawable() != null){  
    ds.writeBytes(lineEnd);  
    BitmapDrawable drawable = (BitmapDrawable) image.getDrawable();  
    Bitmap bitmap = drawable.getBitmap();  
  
    ByteArrayOutputStream baos = new ByteArrayOutputStream();  
    bitmap.compress(Bitmap.CompressFormat.PNG, 100, baos);  
    InputStream fres = new ByteArrayInputStream(baos.toByteArray());  
  
    byte[] buffer = new byte[fres.available()];  
  
    int length = -1;  
    while((length=fres.read(buffer)) != -1){  
        ds.write(buffer,0,length);  
    }  
    ds.writeBytes(lineEnd);  
    ds.writeBytes(lineEnd);  
    ds.writeBytes("--" + boundary + "--" + lineEnd); // requestbody end  
    fres.close();  
}
```

# ITEM

- ❖ ItemRegisterActivity.java 파일에 서버에 데이터를 전송하는 스레드 코드 구현

```
//파일이 없는 경우에는 body의 종료만 생성
else {
    ds.writeBytes(lineEnd);
    ds.writeBytes("--" + boundary + "--" + lineEnd); // requestbody end
}

ds.flush();
ds.close();

//문자열을 다운로드 받기 위한 스트림을 생성
BufferedReader br = new BufferedReader(new InputStreamReader(
    con.getInputStream()));
StringBuilder sb = new StringBuilder();
//문자열을 읽어서 저장
while (true) {
    String line = br.readLine();
    if (line == null)
        break;
    sb.append(line + "\n");
}
```

# ITEM

- ❖ ItemRegisterActivity.java 파일에 서버에 데이터를 전송하는 스레드 코드 구현

```
//사용한 스트림과 연결 해제  
br.close();  
con.disconnect();  
json = sb.toString();  
}catch(Exception e){  
    Log.e("삽입 예외", e.getMessage());  
    message.obj = "삽입 에러로 파라미터 전송에 실패했거나 다운로드 실패\n서버를  
확인하거나 파라미터 전송 부분을 확인하세요";  
    message.what = 0;  
    handler.sendMessage(message);  
}
```

# ITEM

- ❖ ItemRegisterActivity.java 파일에 서버에 데이터를 전송하는 스레드 코드 구현

```
if(json != null){  
  
    try{  
        JSONObject object = new JSONObject(json);  
        String error = object.getString("error");  
        message.obj = error;  
        message.what = 1;  
        handler.sendMessage(message);  
    }catch(Exception e){  
        Log.e("삽입 예외", e.getMessage());  
    }  
}else{  
    Log.e("파싱 실패", "데이터가 포맷에 맞지 않음");  
    message.obj = "파싱 실패";  
    message.what = 0;  
    handler.sendMessage(message);  
}  
}  
}
```

# ITEM

- ❖ ItemRegisterActivity.java 파일의 onCreate 메서드에 버튼 클릭 코드 구현

```
registerbtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        //유효성 검사
        Message message = new Message();
        message.what = 0;
        if(itemnameinput.getText().toString().trim().length() == 0){
            message.obj = "이름은 비어있을 수 없습니다.";
            handler.sendMessage(message);
            return;
        }
        if(priceinput.getText().toString().trim().length() == 0){
            message.obj = "수량은 비어있을 수 없습니다.";
            handler.sendMessage(message);
            return;
        }
        if(descriptioninput.getText().toString().trim().length() == 0){
            message.obj = "설명은 비어있을 수 없습니다.";
            handler.sendMessage(message);
            return;
        }
        new ThreadEx().start();
    }
});
```