

LAB 8 การสร้างแอปพลิเคชันเชื่อมต่อกับ MySQL

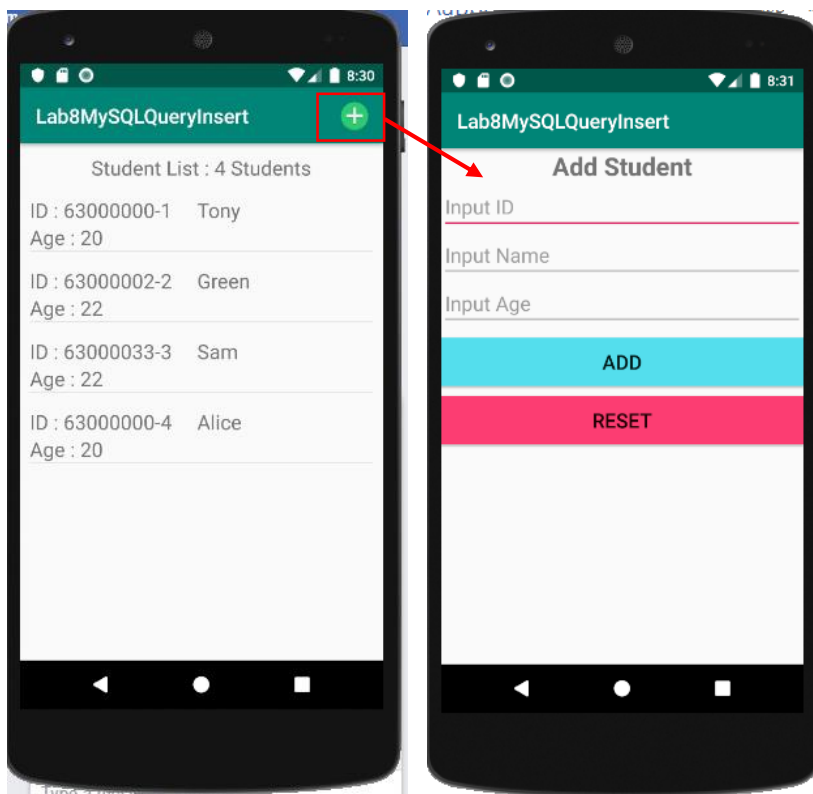
เอกสารประกอบการปฏิบัติการรายวิชา 342267 MOBILE DEVICE PROGRAMMING

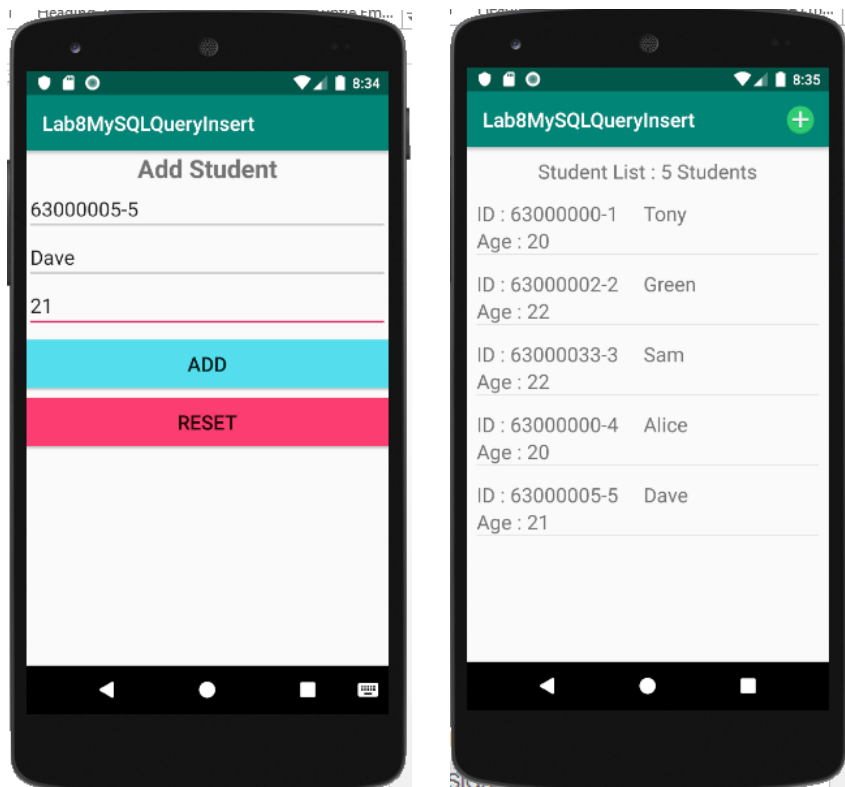
วัตถุประสงค์

1. เพื่อให้นักศึกษาสามารถพัฒนาโปรแกรมให้สามารถเชื่อมต่อกับฐานข้อมูล MySQL ได้
2. เพื่อให้นักศึกษาสามารถพัฒนาโปรแกรมในการ Query และ Insert ข้อมูลในฐานข้อมูล MySQL โดยใช้ NodeJS และ Express

หน้าจอการทำงาน

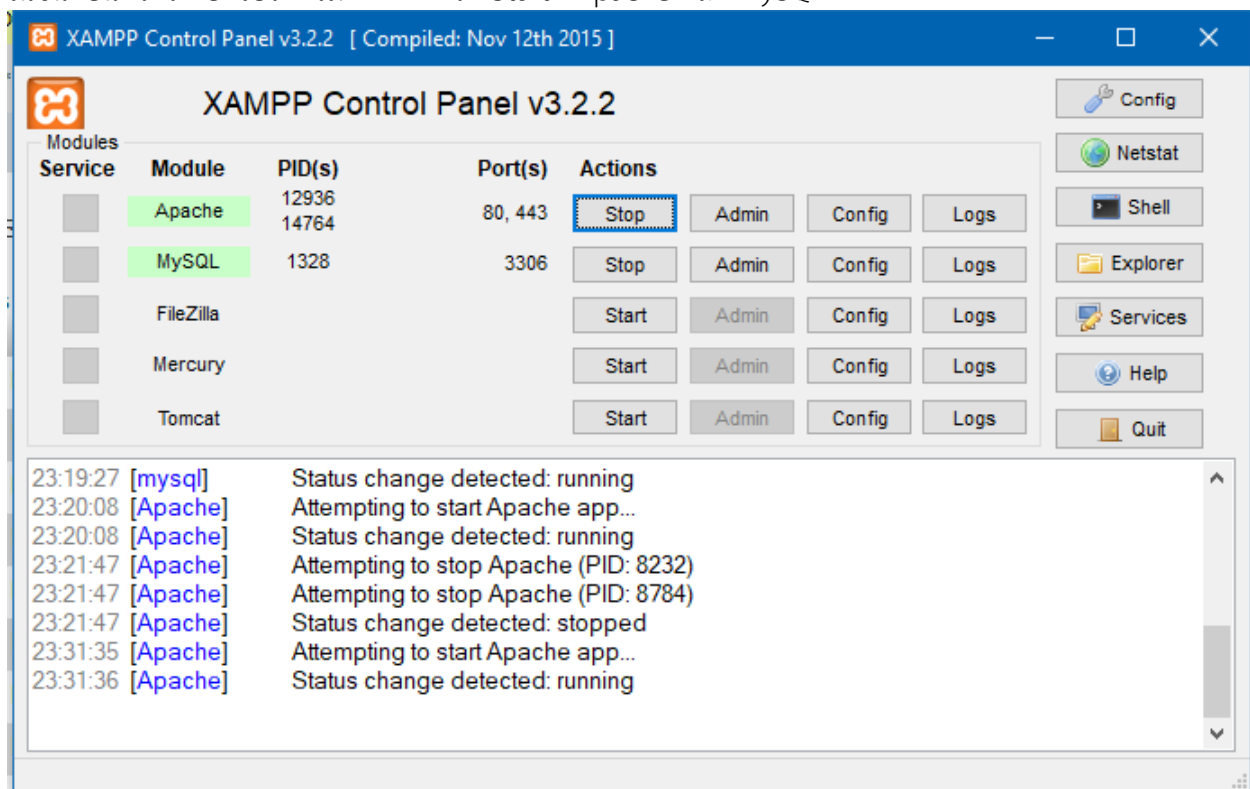
การทำงานหน้าจอแรกจะเป็นการดึงข้อมูลทั้งหมดจากฐานข้อมูลมาแสดง ในหน้าแรกยังมีส่วนที่สามารถเพิ่มข้อมูลลงในฐานข้อมูลได้ แล้วแสดงข้อมูลที่เพิ่มเข้าไปในหน้าจอแรก ดังภาพด้านล่าง



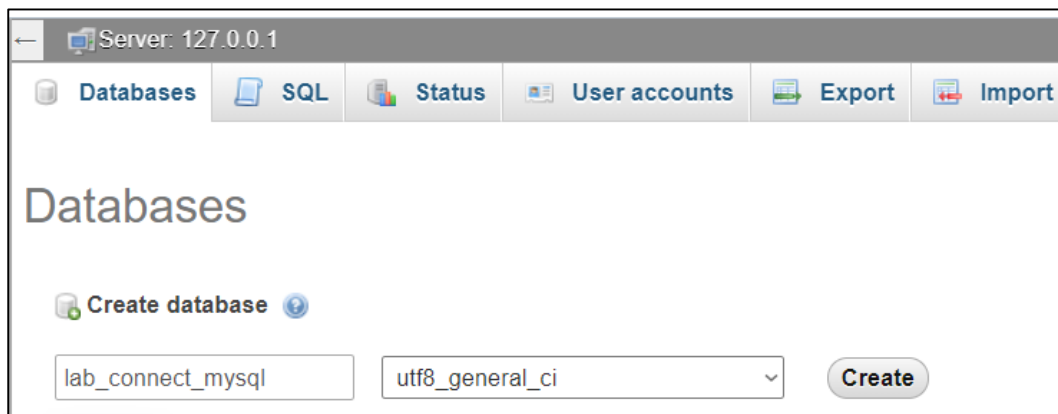


การสร้างฐานข้อมูลที่ MySQL

ในขั้นตอนแรกให้เปิดโปรแกรม XAMPP ให้ Start : Apache และ MySQL



จากนั้นให้เข้าที่ phpMyAdmin เพื่อสร้างฐานข้อมูล โดยให้ชื่อว่า lab_connect_mysql



และให้สร้างตารางชื่อ student ขึ้นมา โดยมี 3 field คือ no, std_id, std_name และ std_age โดยกำหนดให้ no เป็น Primary Key (Auto Increment)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
<input type="checkbox"/> 1	No	int(11)			No	None		AUTO_INCREMENT
<input type="checkbox"/> 2	std_id	varchar(12)	utf8_general_ci		No	None		
<input type="checkbox"/> 3	std_name	varchar(50)	utf8_general_ci		No	None		
<input type="checkbox"/> 4	std_age	int(2)			No	None		

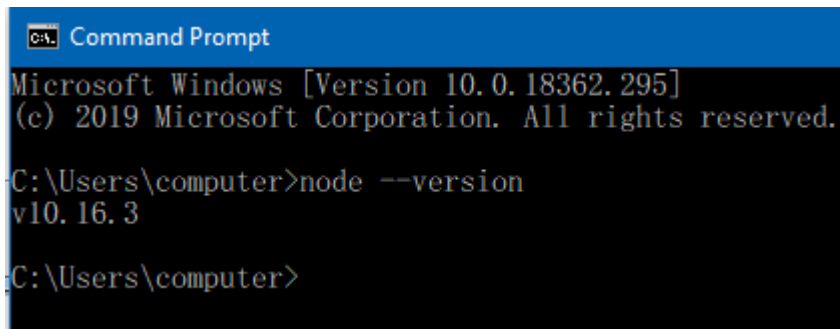
จากนั้นให้เพิ่มข้อมูลเพื่อใช้ในการทดสอบ

No	std_id	std_name	std_age
1	63000000-1	Tony	20
2	63000002-2	Green	22
3	63000033-3	Sam	22

การติดตั้ง NodeJS (เฉพาะเครื่องที่ยังไม่ได้ติดตั้ง)

ให้ Download: **NodeJS** จาก <https://nodejs.org/en/download/> และทำการติดตั้งให้เรียบร้อย

แล้วทำการทดสอบการติดตั้งโดยเปิด CMD แล้วพิมพ์คำสั่ง node -version หากแสดงข้อความดังด้านล่างแสดงว่าติดตั้งได้สำเร็จ



```

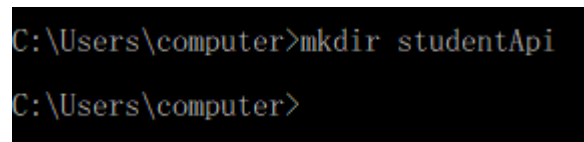
C:\Users\computer>node --version
v10.16.3
C:\Users\computer>

```

การสร้าง Web API โดยใช้ NodeJS

1. ให้สร้างโฟลเดอร์ เพื่อเก็บไฟล์ API โดยพิมพ์คำสั่งใน CMD หรือสร้างเองได้ สำหรับ Lab ตั้งชื่อโฟลเดอร์คือ studentApi

คำสั่ง : mkdir studentApi

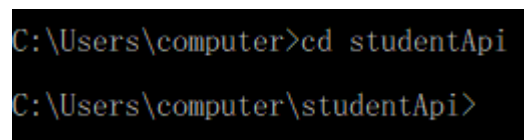


```

C:\Users\computer>mkdir studentApi
C:\Users\computer>

```

2. ให้เข้าไปในโฟลเดอร์ คำสั่ง : cd studentApi

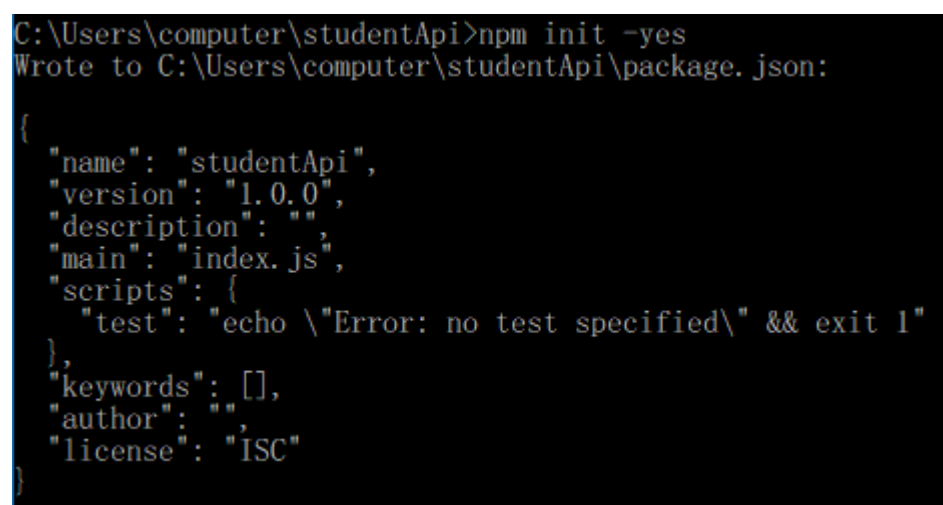


```

C:\Users\computer>cd studentApi
C:\Users\computer\studentApi>

```

3. จากนั้นเริ่มต้นสร้างไฟล์ของ Nodejs ภายในโฟลเดอร์ คำสั่ง: npm init -yes



```

C:\Users\computer\studentApi>npm init -yes
Wrote to C:\Users\computer\studentApi\package.json:

{
  "name": "studentApi",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}

```

4. จากนั้นติดตั้ง Express Framework, mySql, และ body-parser ตามคำสั่ง ดังนี้

npm install express -save

```
C:\Users\computer\studentApi>npm install express -save
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN studentApi@1.0.0 No description
npm WARN studentApi@1.0.0 No repository field.

+ express@4.17.1
added 50 packages from 37 contributors and audited 126 packages in 3.244s
found 0 vulnerabilities
```

npm install mysql -save

```
C:\Users\computer\studentApi>npm install mysql -save
npm WARN studentApi@1.0.0 No description
npm WARN studentApi@1.0.0 No repository field.

+ mysql@2.17.1
added 9 packages from 14 contributors and audited 139 packages in 1.866s
found 0 vulnerabilities
```

npm install body-parser เป็นตัวกลางที่ใช้จัดการ JSON, Raw, Text และ URL encoded ของข้อมูล

```
C:\Users\computer\studentApi>npm install body-parser
npm WARN studentApi@1.0.0 No description
npm WARN studentApi@1.0.0 No repository field.

+ body-parser@1.19.0
updated 1 package and audited 171 packages in 1.464s
found 0 vulnerabilities
```

และสามารถลง nodemon สำหรับคอยติดตามว่าโปรแกรมที่เขียนมีการเปลี่ยนแปลงหรือไม่ ตอนเวลาที่แก้ไขโค้ด โดยที่ไม่ต้องรันโปรแกรมใหม่

npm install -g nodemon

```
PS C:\Users\computer\studentApi> npm install -g nodemon
C:\Users\computer\AppData\Roaming\npm\nodemon -> C:\Users\computer\AppData\Roaming\npm\node_modules\nodemon\bin\nodemon.js

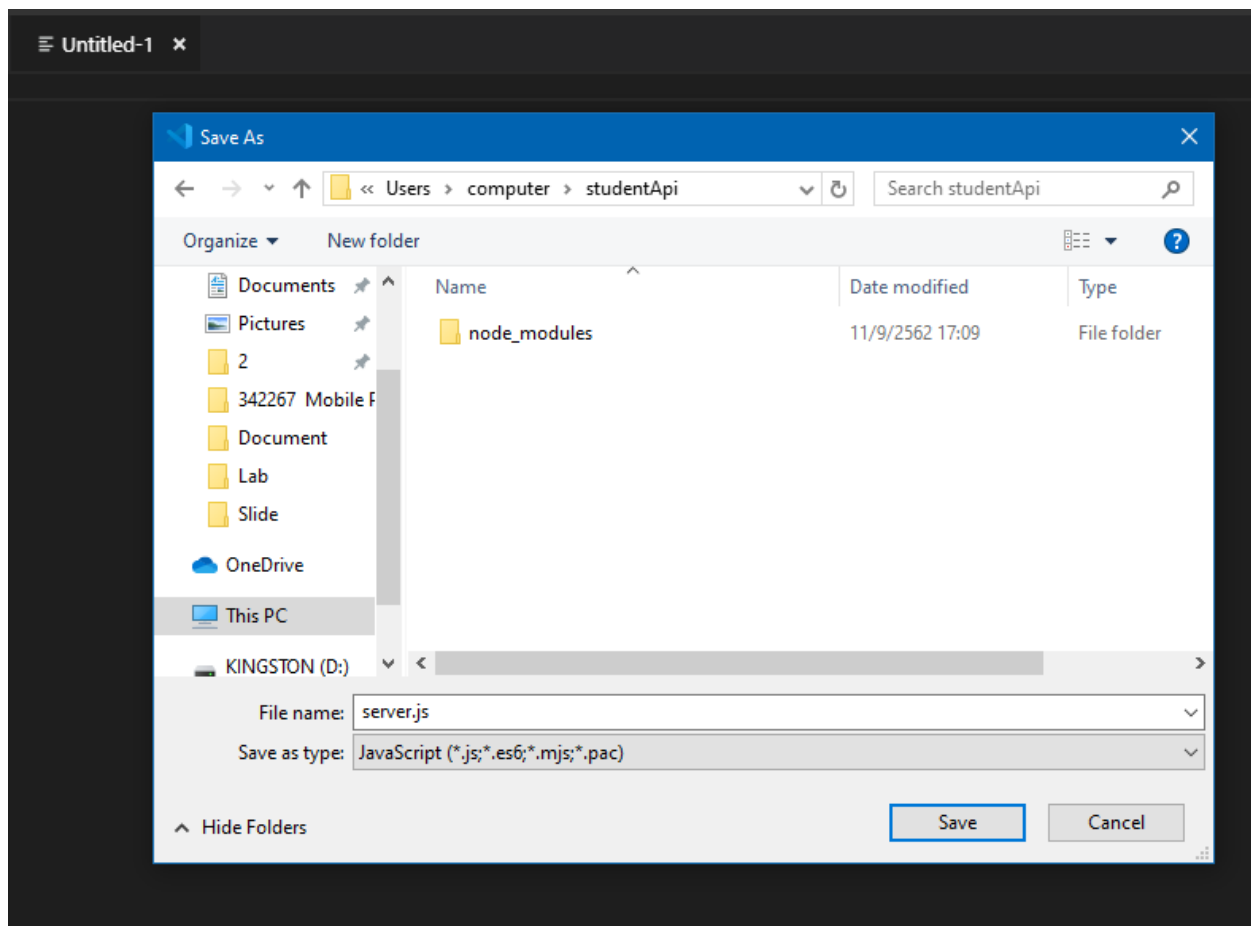
> nodemon@1.19.2 postinstall C:\Users\computer\AppData\Roaming\npm\node_modules\nodemon
> node bin/postinstall || exit 0

npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.9 (node_modules\nodemon\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.9: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

+ nodemon@1.19.2
updated 1 package in 15.116s
```

การสร้างไฟล์ API

1. จากนั้นเปิดโปรแกรม VS Code เพื่อสร้างไฟล์ server.js



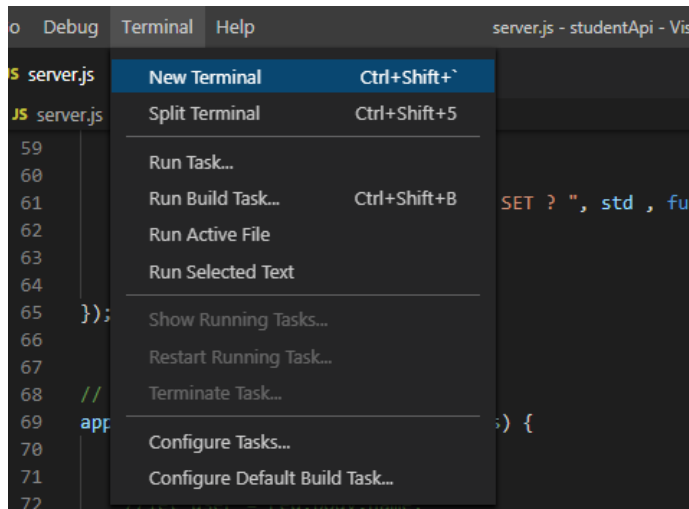
2. จากนั้นพิมพ์คำสั่งสร้าง API

```

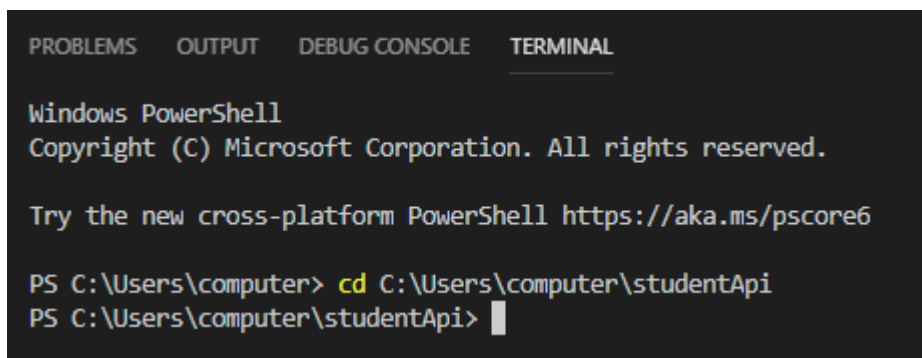
1  var express = require('express');
2  var app = express();
3  var bodyParser = require('body-parser');
4  var mysql = require('mysql');
5
6  app.use(bodyParser.json());
7  app.use(bodyParser.urlencoded({
8    extended: true
9  }));
10
11
12  // default route
13  app.get('/', function (req, res) {
14    return res.send({ error: true, message: 'Test Student Web API' });
15  });
16  // connection configurations
17  var dbConn = mysql.createConnection({
18    host: 'localhost',
19    user: 'root',
20    password: '',
21    database: 'lab_connect_mysql'
22  });
23
24  // connect to database
25  dbConn.connect();
26
27  // Retrieve all students
28  app.get('/allstd', function (req, res) {
29    dbConn.query('SELECT * FROM student', function (error, results, fields) {
30      if (error) throw error;
31      return res.send(results);
32    });
33  });
34
35
36  // Add a new Student
37  app.post('/std', function (req, res) {
38
39    var std = req.body
40
41    if (!std) {
42      return res.status(400).send({ error:true, message: 'Please provide student ' });
43    }
44
45    dbConn.query("INSERT INTO student SET ? ", std , function (error, results, fields) {
46      if (error) throw error;
47      return res.send(results);
48    });
49  });
50
51  // set port
52  app.listen(3000, function () {
53    console.log('Node app is running on port 3000');
54  });
55
56  module.exports = app;

```

3. จากนั้นให้เปิด Terminal

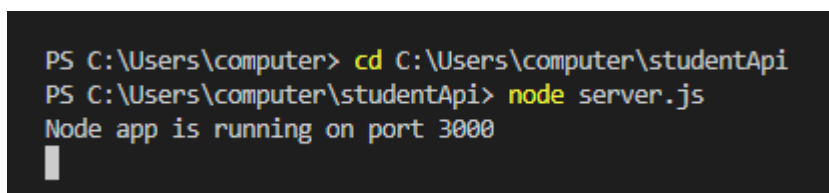


3. ถัดมาพิมพ์คำสั่งเพื่อเข้าไปที่โฟลเดอร์ที่ได้สร้างไว้ เช่น `cd C:\Users\computer\studentApi`



4. ให้ Run Server เพื่อเปิด port รอให้บริการ

node server.js



หรือ nodemon server.js


```

PS C:\Users\computer\studentApi> nodemon server.js
[nodemon] 1.19.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching dir(s): *.*
[nodemon] starting `node server.js`
Node app is running on port 3000

```

การใช้ postman ในการทดสอบ API

แสดงข้อมูลทั้งหมดของนักศึกษา

ให้กำหนดวิธีส่งข้อมูลเป็น Get และใส่ URL เป็น 127.0.0.1:3000/allstd จากนั้นกดปุ่ม Send

The screenshot shows the Postman interface with a GET request to `127.0.0.1:3000/allstd` sent successfully. The status bar indicates `Status: 200 OK`, `Time: 29ms`, and `Size: 400 B`. The response body is displayed in JSON format, showing an array of three student objects.

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body | Cookies | Headers (6) | Test Results

Status: 200 OK Time: 29ms Size: 400 B Sav

Pretty | Raw | Preview | Visualize BETA | JSON

```

1  [
2    {
3      "No": 1,
4      "std_id": "63000000-1",
5      "std_name": "Tony",
6      "std_age": 20
7    },
8    {
9      "No": 2,
10     "std_id": "63000002-2",
11     "std_name": "Green",
12     "std_age": 22
13   },
14   {
15     "No": 3,
16     "std_id": "63000033-3",
17     "std_name": "Sam",
18     "std_age": 22
19   }
20 ]

```

แสดงผลลัพธ์ที่ตอบกลับมา

การเพิ่มข้อมูล

ให้กำหนดวิธีส่งข้อมูลเป็น Post และใส่ URL เป็น 127.0.0.1:3000/std และในส่วนของ body จะมีการกำหนดค่าเพื่อส่งข้อมูลในการเพิ่มในตาราง student และจะต้องกำหนด Content-type เป็น JSON(application/json) จากนั้นกดปุ่ม Send ดังนี้

The screenshot shows a REST client interface with the following details:

- Method:** POST
- URL:** 127.0.0.1:3000/std/
- Body Type:** raw (selected), with a dropdown menu showing options like form-data, x-www-form-urlencoded, raw, binary, GraphQL BETA, and JSON.
- Request Body (JSON):**

```
{
  "std_id": "63000000-4",
  "std_name": "Alice",
  "std_age": 20
}
```
- Status:** 200 OK
- Response Body (JSON):**

```
{
  "fieldCount": 0,
  "affectedRows": 1,
  "insertId": 30,
  "serverStatus": 2,
  "warningCount": 0,
  "message": "",
  "protocol141": true,
}
```

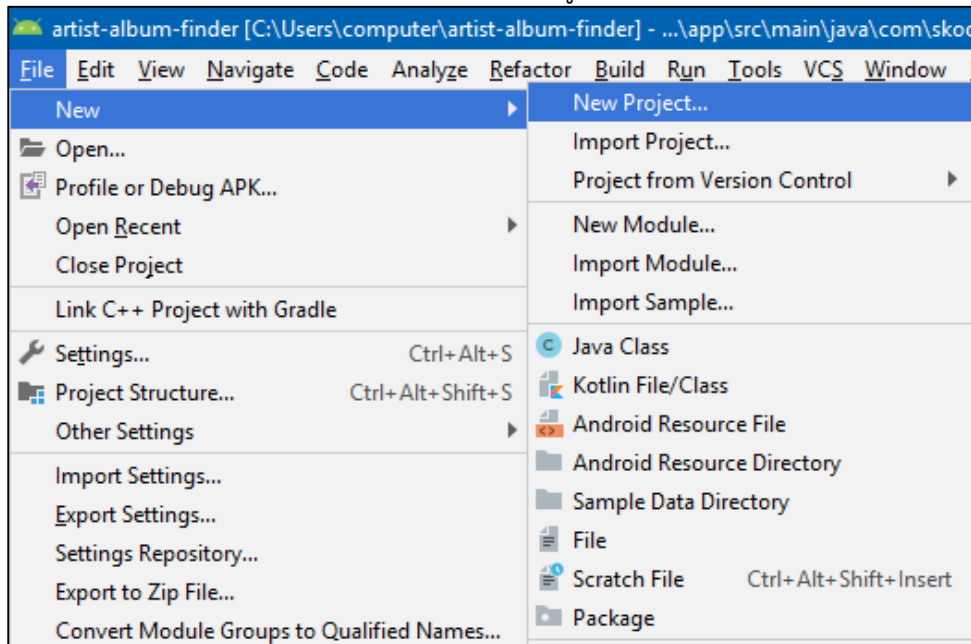
A red box highlights the response status "Status: 200 OK". A red arrow points from a text box labeled "แสดงผลลัพธ์ที่ตอบกลับมา" (Show the result that comes back) to the response body.

เมื่อเข้าไปดูข้อมูลใน phpMyadmin จะเห็นมีข้อมูลที่เพิ่มขึ้นมา

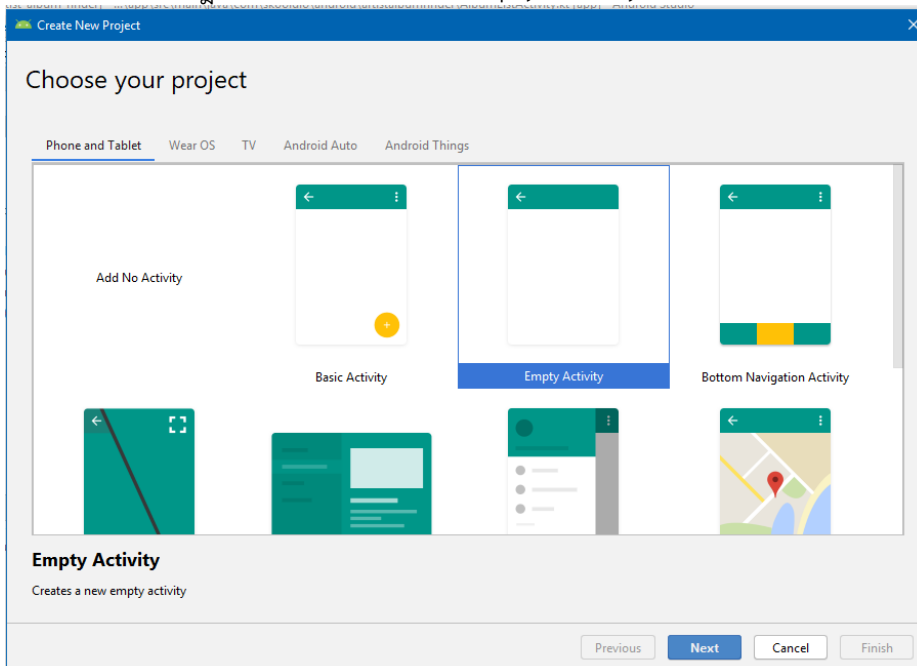
No	std_id	std_name	std_age
1	63000000-1	Tony	20
2	63000002-2	Green	22
3	63000033-3	Sam	22
4	63000000-4	Alice	20

การเขียนคำสั่งในส่วนของ Android

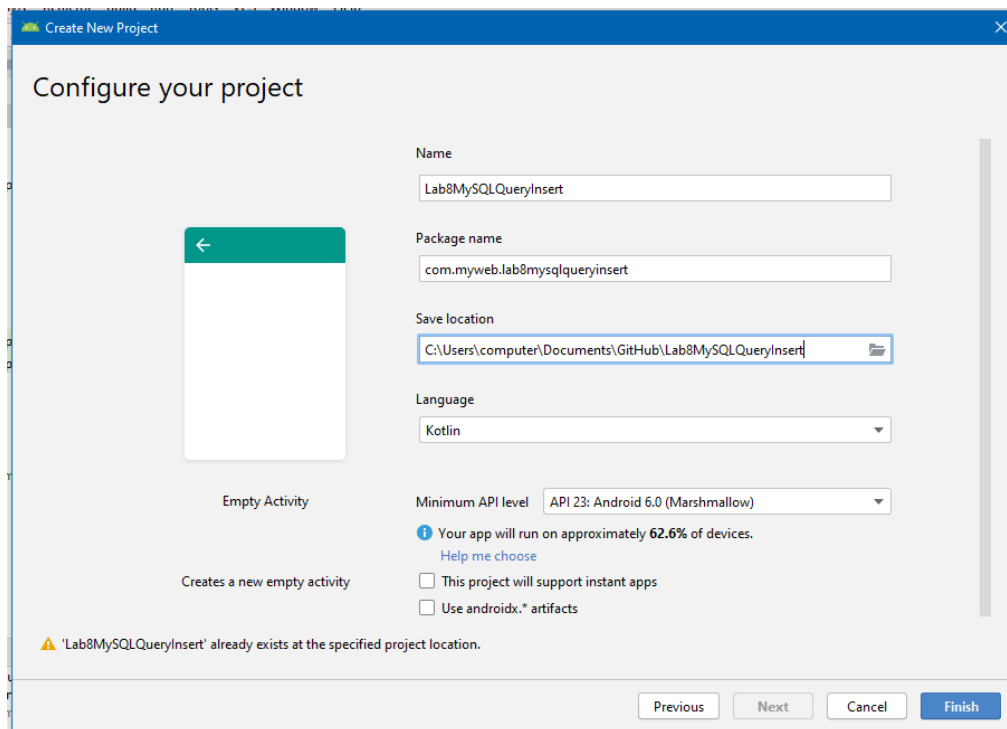
1. เมื่อเปิดโปรแกรม Android Studio แล้วไปที่เมนู File แล้วเลือก New คลิกที่ New Project



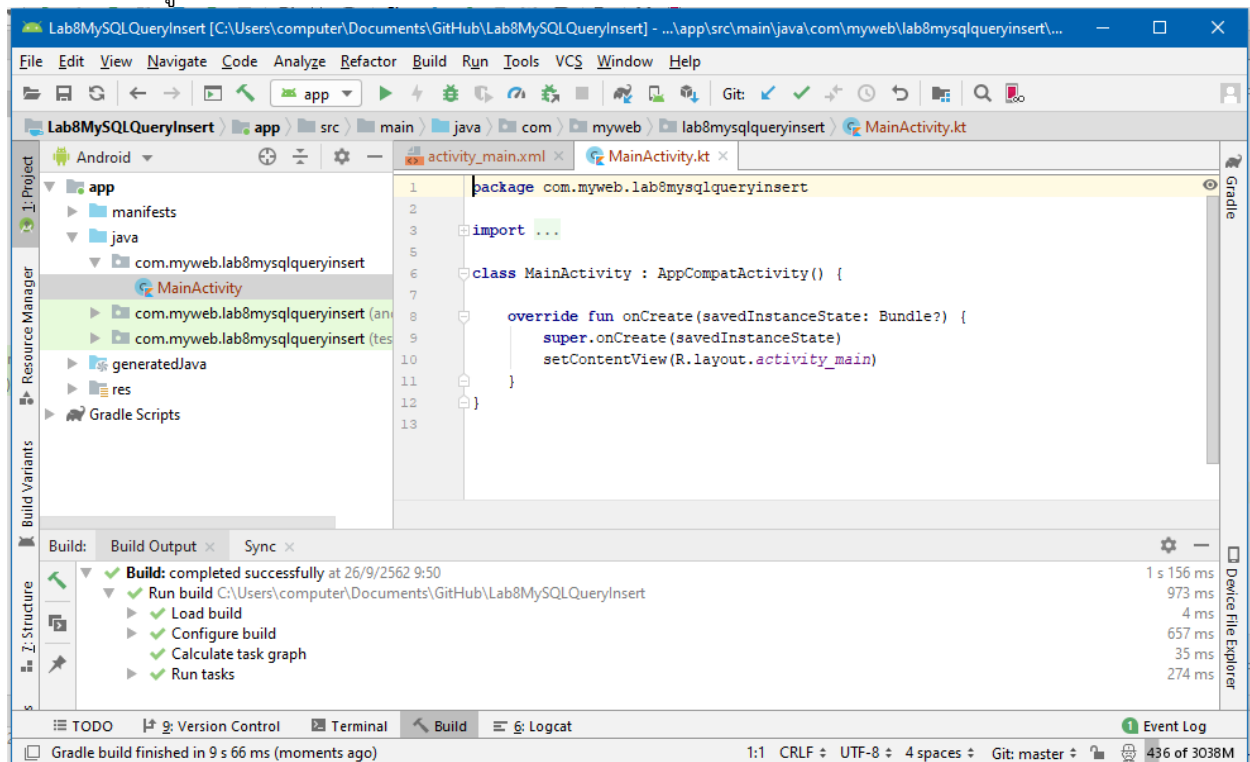
2. จากนั้นจะปรากฏหน้าจอ ดังนี้ ให้เลือก Empty Activity จากนั้นกด Next



3. ถัดมาให้ตั้งชื่อ Application name คือ Lab8MySQLQueryInsert และ Language เป็น Kotlin จากนั้นกด Finish

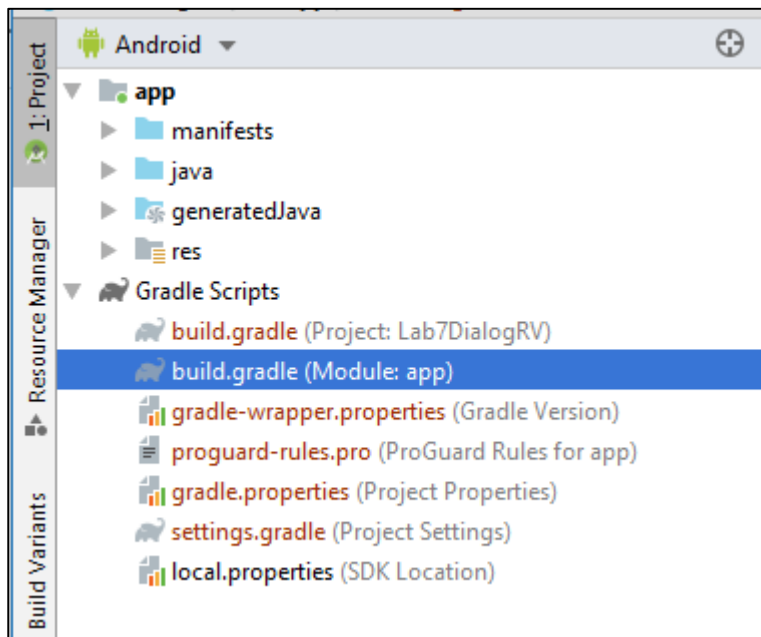


จะต้องรอสักครู่ เพื่อให้โปรแกรมสร้างโปรเจกเสร็จ



การเพิ่ม Library ของไฟล์ app.Gradle

1. ให้ไปที่ Gradle Scripts คลิกที่ build.gradle(Module:app)



2. ให้เพิ่มคำสั่งดังนี้

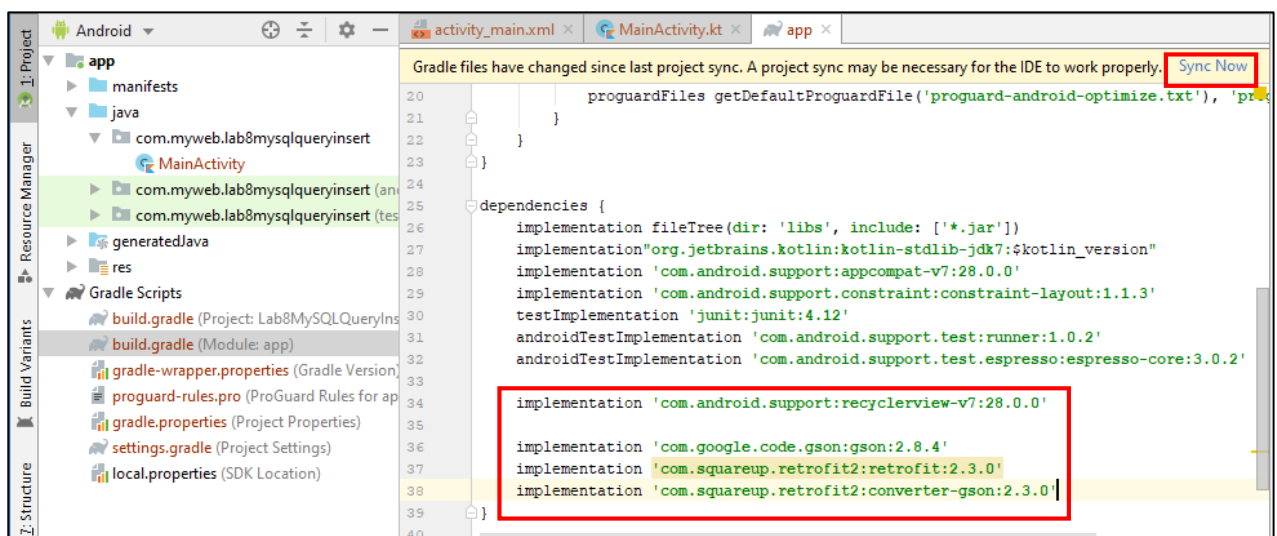
```
implementation 'com.google.code.gson:gson:2.8.4'
```

```
implementation 'com.squareup.retrofit2:retrofit:2.3.0'
```

```
implementation 'com.squareup.retrofit2:converter-gson:2.3.0'
```

```
implementation 'com.android.support:recyclerview-v7:28.0.0'
```

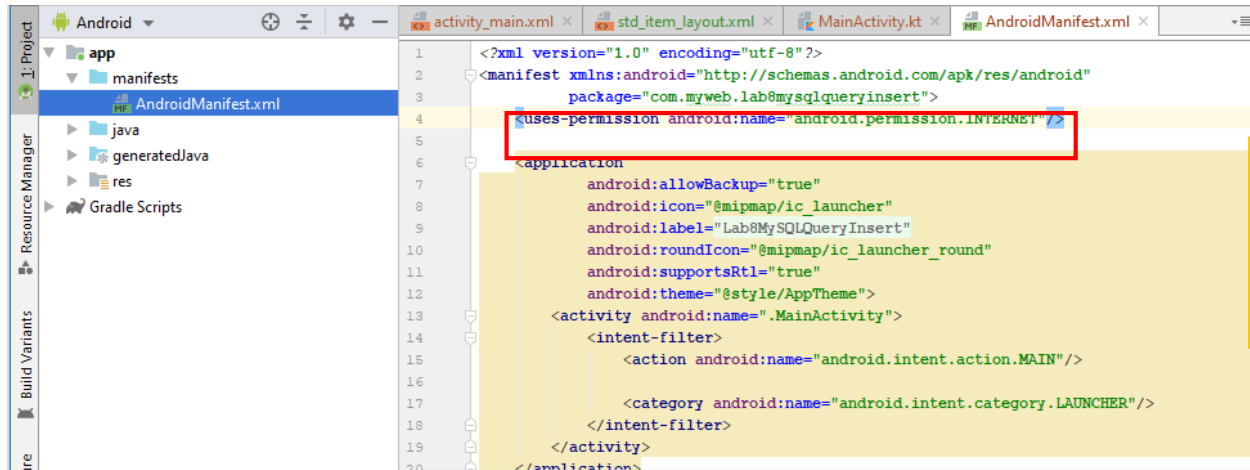
จากนั้นให้คลิกที่ Sync Now ด้านบนขวา เพื่อโหลด Library ของ RecyclerView, Retrofit และ Gson



ส่วนของไฟล์ AndroidManifest.xml

ให้เข้าไปที่ app >> manifests >> AndroidManifest.xml จากนั้นเพิ่มคำสั่ง

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



โครงสร้างของโปรเจค

การออกแบบ Layout

option_menu.xml >> Option menu

activity_main.xml >> หน้า main

std_item_laout >> การออกแบบส่วน item ของ RecyclerView

activity_insert.xml >> หน้าจอเพิ่มข้อมูล

การสร้างการทำงาน

Student.kt >> Data Class ของนักศึกษา

StudentsAdapter.kt >> Adapter ที่ทำงานกับRecyclerView

StudentAPI.kt >> interface ทำงานกับ API

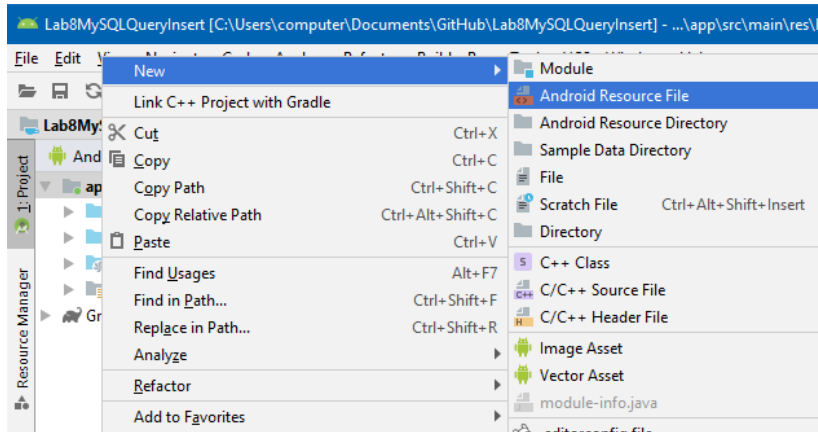
InsertActivity.kt >> การเพิ่มข้อมูล

MainActivity.kt >> Class สำหรับจัดการเกี่ยวกับ UI

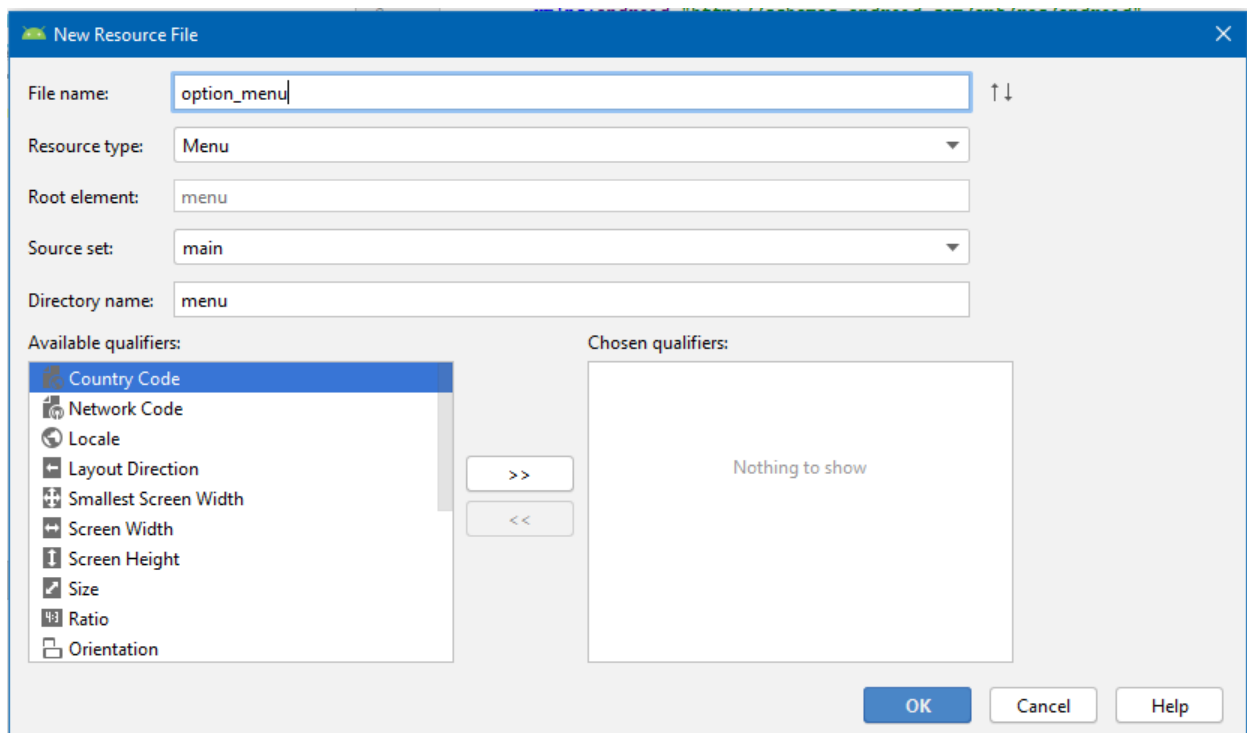
การออกแบบ Layout

การสร้าง Option Menu

1. ให้ไปที่ File >> New >> Android Resource File



2. ให้ตั้งชื่อไฟล์ option_menu และกำหนด Resource type เป็น Menu แล้วคลิกที่ปุ่ม OK



3. ให้เพิ่มคำสั่ง ดังนี้ สำหรับรูป icon ให้เก็บไว้ที่โฟลเดอร์

Lab8MySQLQueryInsert\app\src\main\res\drawable

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <menu xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto">
4      <item android:id="@+id/item1"
5          android:title="Insert Student"
6          android:icon="@drawable/sign_add_icon"
7          app:showAsAction="always"/>
8  </menu>

```

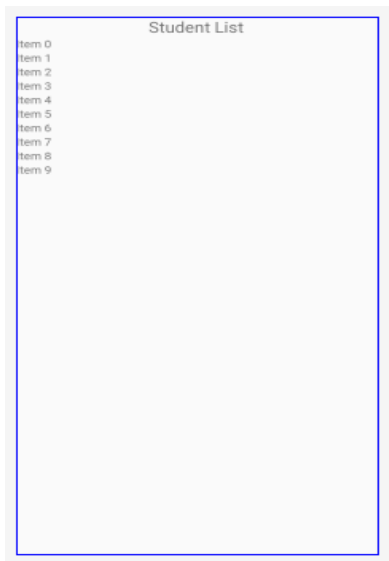
การเพิ่มคำสั่ง ไฟล์ activity_main.xml ดังนี้

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:tools="http://schemas.android.com/tools"
5      xmlns:app="http://schemas.android.com/apk/res-auto"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      android:orientation="vertical"
9      tools:context=".MainActivity">
10
11      <TextView
12          android:id="@+id/text1"
13          android:layout_width="match_parent"
14          android:layout_height="wrap_content"
15          android:text="Student List"
16          android:textAlignment="center"
17          android:textSize="20sp" />
18
19      <android.support.v7.widget.RecyclerView
20          android:id="@+id/recycler_view"
21          android:layout_width="match_parent"
22          android:layout_height="match_parent"
23          android:scrollbars="vertical"/>
24
25  </LinearLayout>

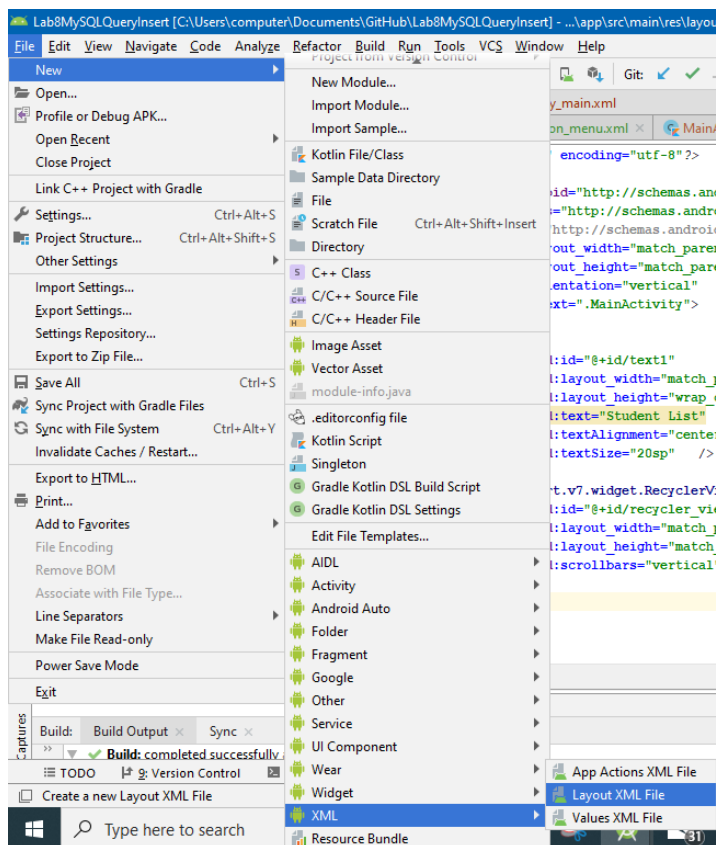
```


ส่วนของ design จะแสดงดังนี้

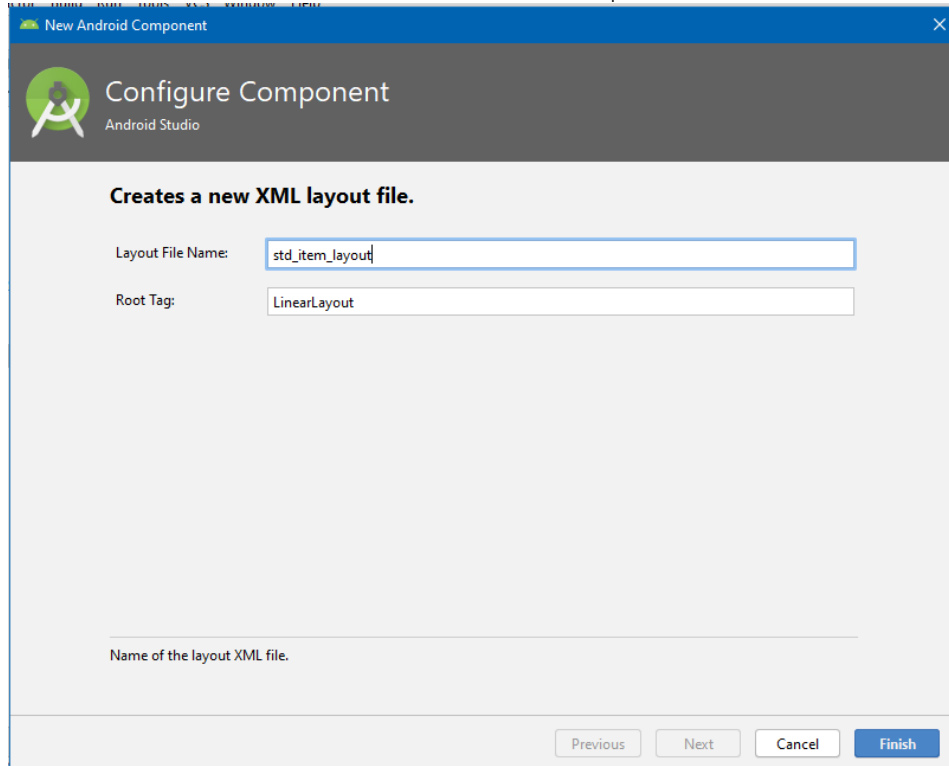


การเพิ่มคำสั่งในไฟล์ std_item_laout.xml ดังนี้

1. ให้ไปที่ File >> New >> XML >> Layout XML File



2. จากนั้นให้พิมพ์ชื่อ layout คือ std_item_layout แล้วคลิกปุ่ม Finish



หลังจากนั้นให้เพิ่มคำสั่ง ดังภาพด้านล่าง

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <RelativeLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      android:layout_width="match_parent"
5      android:layout_height="wrap_content">
6      <TextView
7          android:id="@+id/tv_id"
8          android:layout_width="wrap_content"
9          android:layout_height="wrap_content"
10         android:layout_alignParentTop="true"
11         android:layout_marginTop="15dp"
12         android:text="ID"
13         android:textSize="20sp" />
14     <TextView
15         android:id="@+id/tv_name"
16         android:layout_width="wrap_content"
17         android:layout_height="wrap_content"
18         android:textSize="20sp"
19         android:text="Name"
20         android:layout_marginTop="15dp"
21         android:layout_marginLeft="20dp"
22         android:layout_toRightOf="@id/tv_id" />

```

```

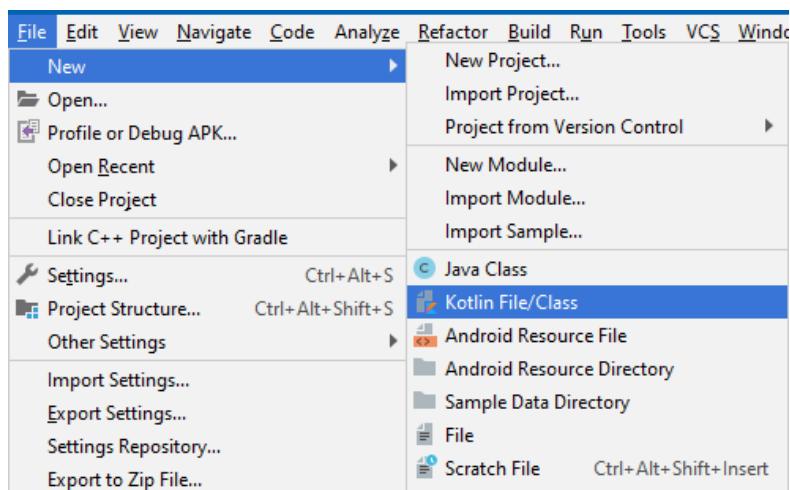
23      <TextView
24          android:id="@+id/tv_age"
25          android:layout_width="wrap_content"
26          android:layout_height="wrap_content"
27          android:text="Age"
28          android:textSize="20sp"
29          android:layout_below="@id/tv_name" />
30  </RelativeLayout>

```

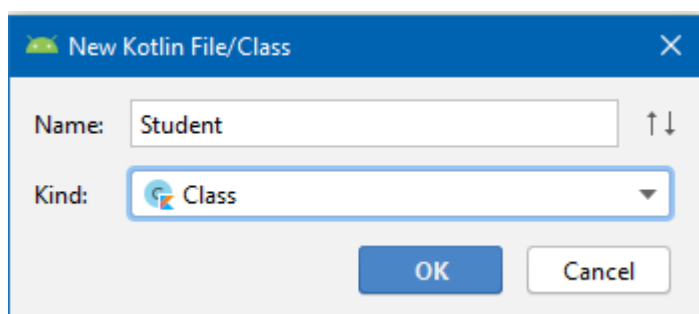
การสร้างการทำงานส่วน File Kotlin

สร้าง Data Class ของ Student.kt

1. ให้ไปที่ File >> New >> Kotlin File/Class



2. จากนั้นจะแสดงหน้าจอให้กรอกชื่อคลาสชื่อ Student และกำหนดชนิดไฟล์เป็น Class

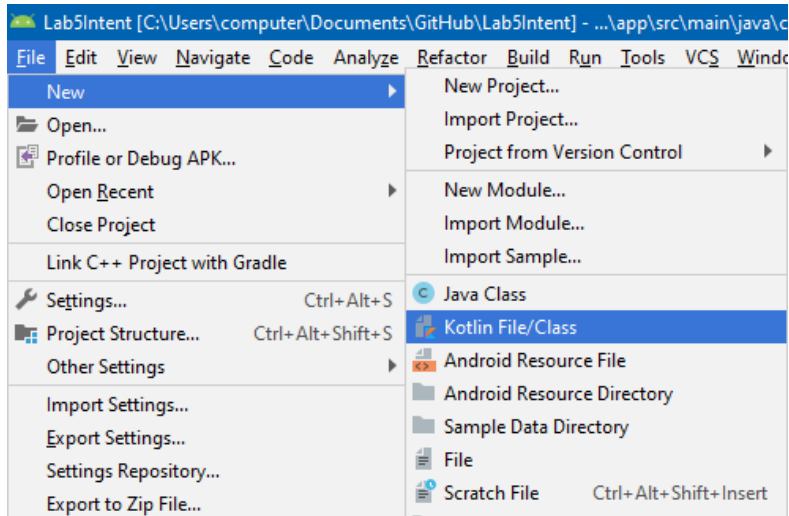


3. แล้วให้พิมพ์โค้ดคำสั่ง โดยกำหนดให้เป็น data class และให้มี รหัสนักศึกษา (id), ชื่อนักศึกษา (name) เป็นประเภท String และอายุ (age) เป็นประเภท Integer

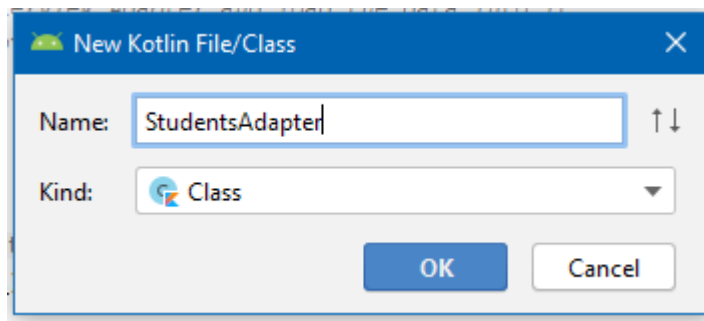
```
1 package com.myweb.lab8mysqlqueryinsert
2
3 import com.google.gson.annotations.Expose
4 import com.google.gson.annotations.SerializedName
5
6 data class Student(
7     @Expose
8     @SerializedName(value: "std_id") val std_id: String,
9
10    @Expose
11    @SerializedName(value: "std_name") val std_name: String,
12
13    @Expose
14    @SerializedName(value: "std_age") val std_age: Int)
```

การสร้างไฟล์ StudentsAdapter.kt (ใช้คำสั่งเดิมจาก Lab 7)

1. ให้ไปที่ File >> New >> Kotlin File/Class

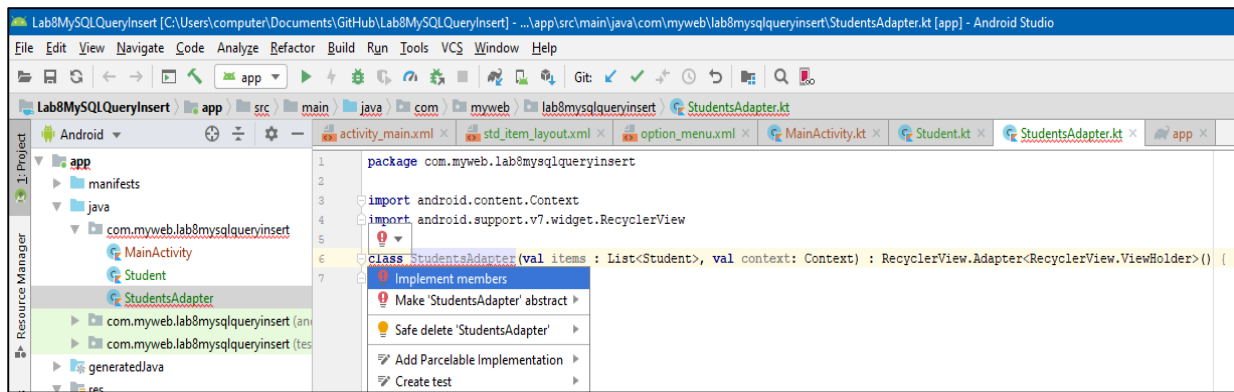


2. จากนั้นจะแสดงหน้าจอให้กรอกชื่อคลาสชื่อ StudentsAdapter

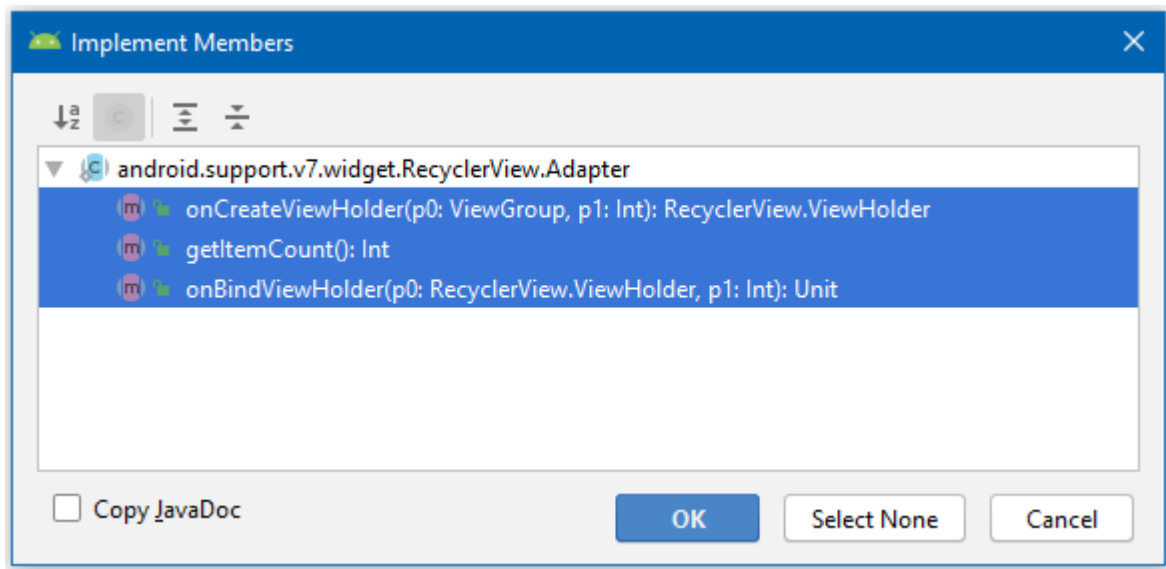


3. สำหรับ Class StudentsAdapter ทำการ extends RecyclerView.Adapter <StudentAdapter. ViewHolder> โดยสร้างตัวแปรประเภท List<Student> และ Context แล้วกำหนดค่าผ่าน constructor

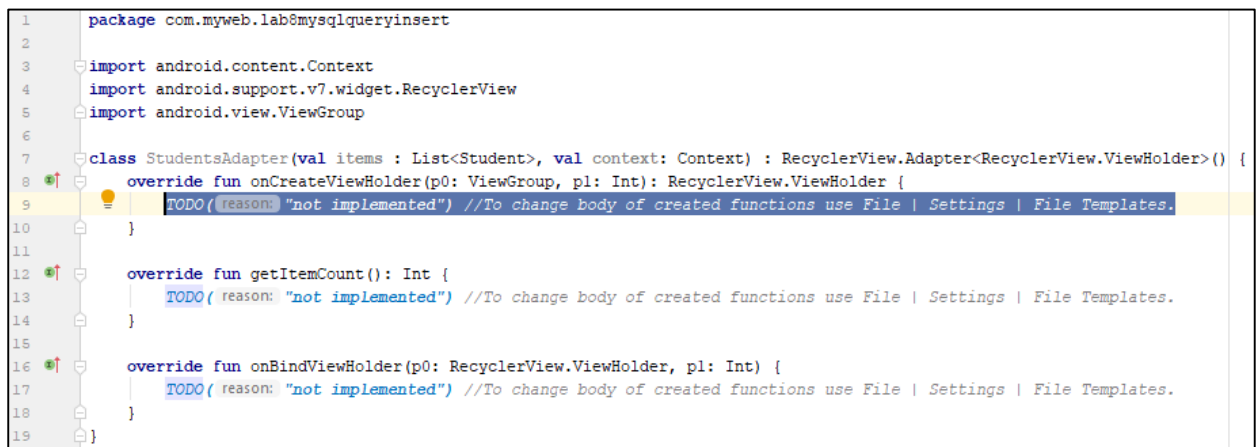
เมื่อแสดงข้อความ Error ให้คลิกที่  แล้วเลือก Implement members



จากนั้นจะขึ้นหน้าต่างให้เลือกทุก Function แล้วคลิกปุ่ม OK



แสดงหน้าจอดังนี้



หลังจากนั้นให้เพิ่มคำสั่งสร้าง class “ViewHolder” extends RecyclerView.ViewHolder ในส่วนด้านล่างสุด
ดังภาพด้านล่าง

```

9  class StudentsAdapter(val items : List<Student>, val context: Context) : RecyclerView.Adapter<RecyclerView.ViewHolder>() {
10
11  override fun onCreateViewHolder(p0: ViewGroup, p1: Int): RecyclerView.ViewHolder {
12      TODO( reason: "not implemented") //To change body of created functions use File | Settings | File Templates.
13  }
14  override fun getItemCount(): Int {
15
16      TODO( reason: "not implemented") //To change body of created functions use File | Settings | File Templates.
17  }
18
19  override fun onBindViewHolder(p0: RecyclerView.ViewHolder, p1: Int) {
20      TODO( reason: "not implemented") //To change body of created functions use File | Settings | File Templates.
21  }
22  }
23
24  class ViewHolder (view: View) : RecyclerView.ViewHolder(view) {
25      // Holds the TextView that will add each student to
26      val tvID :TextView! = view.tv_id
27      val tvName :TextView! = view.tv_name
28      val tvAge :TextView! = view.tv_age
29  }

```

จากนั้นให้แก้ไขคำสั่งด้านบนของการส่งค่ากลับของ Class “StudentsAdapter” และส่วนของฟังก์ชัน onCreateViewHolder, getItemCount, onBindViewHolder ใน Class ของ StudentsAdapter ดังนี้

```

class StudentsAdapter(val items : List<Student>, val context: Context) : RecyclerView.Adapter<ViewHolder>() {
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {
        val view_item = LayoutInflater.from(parent.context).inflate(R.layout.std_item_layout, parent, attachToRoot: false)
        return ViewHolder(view_item)
    }

    override fun getItemCount(): Int {
        return items.size
    }

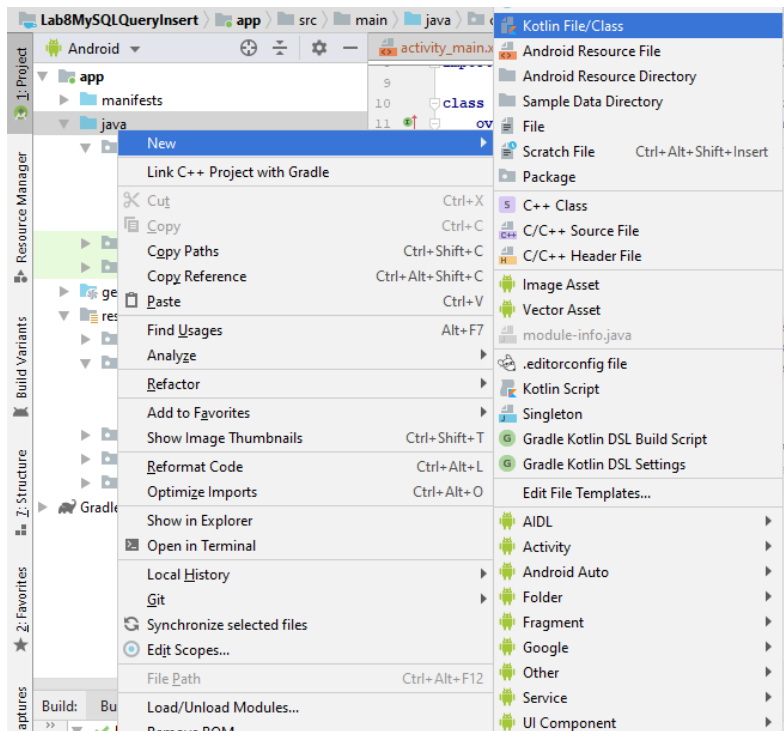
    override fun onBindViewHolder(holder: ViewHolder, position: Int) {
        holder.tvID?.text = items[position].std_id
        holder.tvName?.text = items[position].std_name
        holder.tvAge?.text = items[position].std_age.toString()
    }
}

class ViewHolder (view: View) : RecyclerView.ViewHolder(view) {
    // Holds the TextView that will add each student to
    val tvID = view.tv_id
    val tvName = view.tv_name
    val tvAge = view.tv_age
}

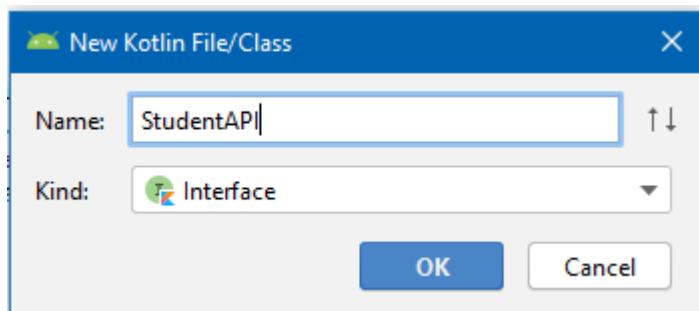
```

การสร้างไฟล์ StudentAPI.kt สำหรับ interface ทำงานกับ API

1. ให้ไปที่ File >> New >> Kotlin File/Class



2. จากนั้นจะแสดงหน้าจอให้กรอกชื่อคลาสชื่อ StudentAPI และ Kind เป็น Interface แล้วคลิกที่ปุ่ม OK



3. สำหรับ Interface ชื่อ StudentAPI ให้เพิ่มคำสั่ง ดังนี้

```

9  interface StudentAPI {
10     @GET( value: "allstd")
11     fun retrieveStudent(): Call<List<Student>>
12
13
14     @FormUrlEncoded
15     @POST( value: "std")
16     fun insertStd(
17         @Field( value: "std_id") std_id: String,
18         @Field( value: "std_name") std_name: String,
19         @Field( value: "std_age") std_age: Int): Call<Student>
20 }
  
```

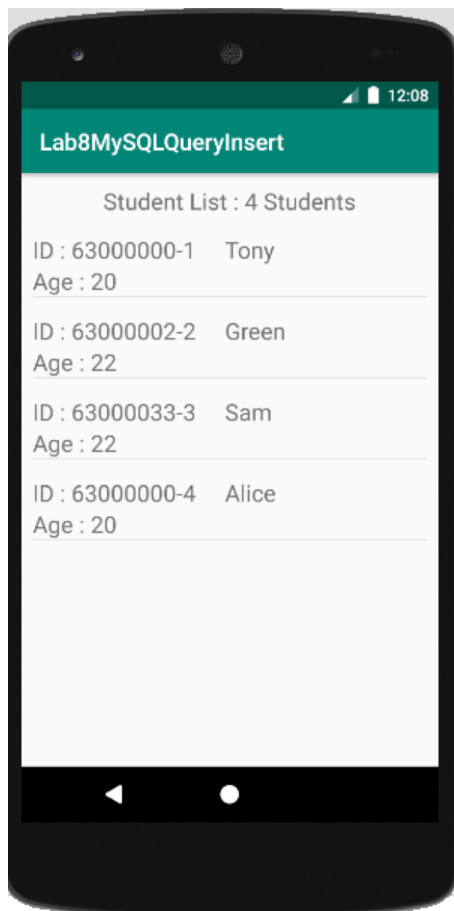

การเพิ่มคำสั่งในไฟล์ MainActivity.kt

```

21 class MainActivity : AppCompatActivity() {
22     var studentList : ArrayList<Student> = arrayListOf<Student>()
23     override fun onCreate(savedInstanceState: Bundle?) {
24         super.onCreate(savedInstanceState)
25         setContentView(R.layout.activity_main)
26
27         // Link to RecyclerView
28         recycler_view.layoutManager = LinearLayoutManager(applicationContext) as RecyclerView.LayoutManager?
29         recycler_view.itemAnimator = DefaultItemAnimator() as RecyclerView.ItemAnimator?
30         recycler_view.addItemDecoration(DividerItemDecoration(recycler_view.getContext(), DividerItemDecoration.VERTICAL))
31
32         recycler_view.setOnItemClickListener(object : OnItemClickListener{
33             override fun onItemClick(position: Int, view: View) {
34                 Toast.makeText(applicationContext, text: "You click on : "+studentList[position].std_id,
35                     Toast.LENGTH_SHORT).show()
36             }
37         })
38     }
39
40     override fun onResume() {
41         super.onResume()
42         callStudentData()
43     }
44
45     fun callStudentData(){
46         studentList.clear();
47
48         val serv : StudentAPI = Retrofit.Builder()
49             .baseUrl( baseUrl: "http://10.0.2.2:3000/")
50             .addConverterFactory(GsonConverterFactory.create())
51             .build()
52             .create(StudentAPI ::class.java)
53
54         serv.retrieveStudent()
55             .enqueue(object : Callback<List<Student>> {
56                 override fun onResponse(call: Call<List<Student>>, response: Response<List<Student>>) {
57                     response.body()?.forEach { it: Student
58                         studentList.add(Student(it.std_id, it.std_name, it.std_age))
59                     }
60                     //// Set Data to RecyclerView
61                     recycler_view.adapter = StudentsAdapter(studentList, applicationContext)
62                     text1.text = "Student List : "+ studentList.size.toString() + " Students"
63                 }
64                 override fun onFailure(call: Call<List<Student>>, t: Throwable) : Unit = t.printStackTrace()
65             })
66     }
67
68 }
69
70
71 interface OnItemClickListener {
72     fun onItemClick(position: Int, view: View)
73 }
74 fun RecyclerView.setOnItemClickListener(onClickListener: OnItemClickListener) {
75     this.addChildAttachStateChangeListener(object: RecyclerView.OnChildAttachStateChangeListener {
76         override fun onChildViewDetachedFromWindow(view: View) {
77             view?.setOnItemClickListener(null)
78         }
79     })
80
81     override fun onChildViewAttachedToWindow(view: View) {
82         view?.setOnItemClickListener { it: View!
83             val holder : RecyclerView.ViewHolder! = getChildViewHolder(view)
84             onClickListener.onItemClick(holder.adapterPosition, view)
85         }
86     }
87 }
88

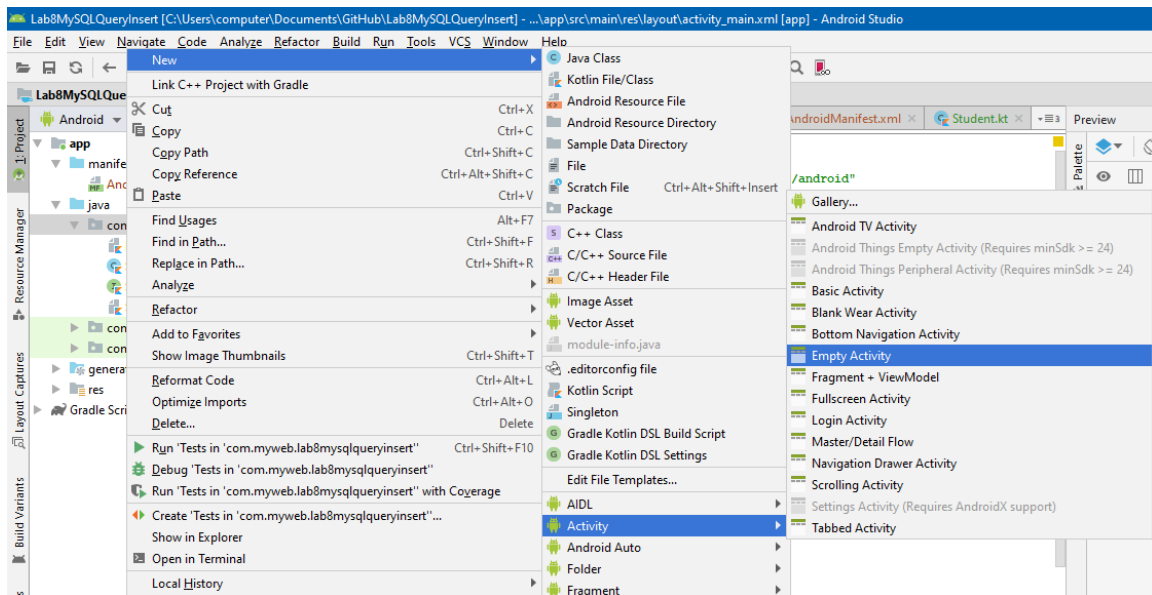
```

จากนั้นให้ทดลอง Run โปรแกรม จะแสดงผลลัพธ์ ดังนี้

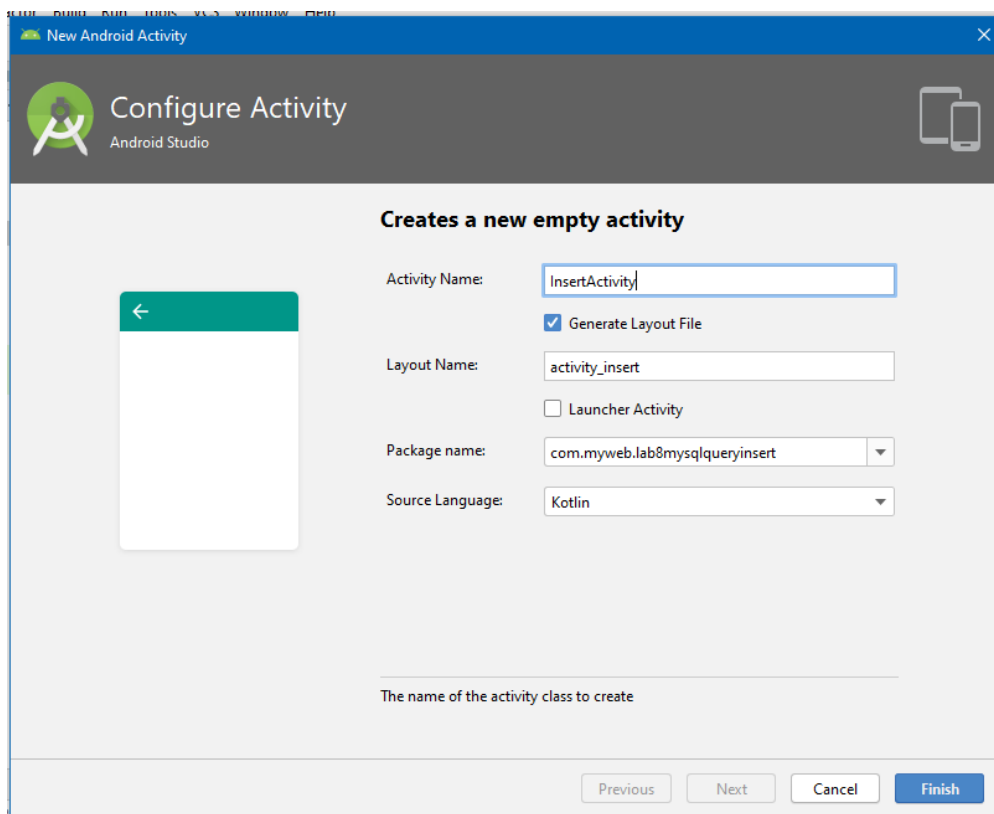


การสร้างหน้าจอเพิ่มข้อมูลนักศึกษา

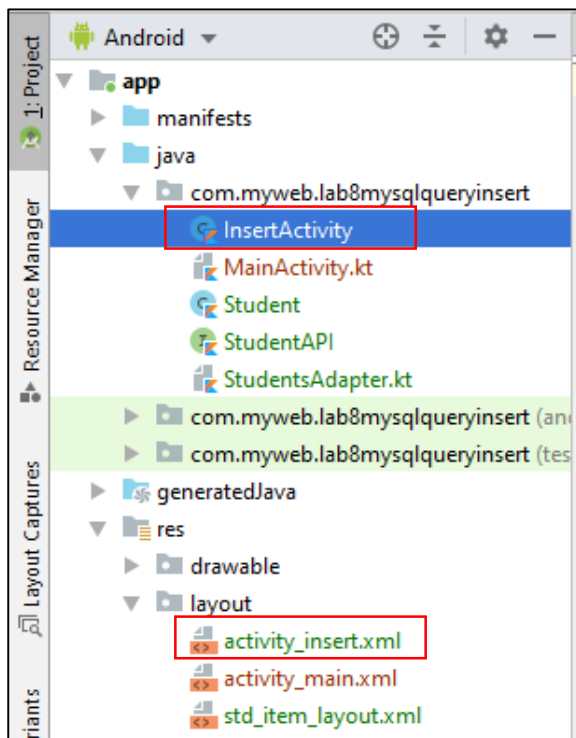
1. ให้สร้าง Activity ใหม่ขึ้นมา ชื่อ InsertActivity ให้ไปที่ File >> New >> Activity >> เลือก Empty Activity



2. จากนั้นให้กรอก Activity Name คือ InsertActivity และต้องเลือก Source Language เป็น Kotlin แล้วกดปุ่ม Finish



3. จากนั้นโปรแกรมจะสร้างไฟล์ให้ 2 ไฟล์ คือ InsertActivity.kt และ activity_insert.xml



การสร้างหน้าจอของ activity_insert.xml

1. หน้าจอ activity_insert จะเป็นการรับค่าจากผู้ใช้ เพื่อนำข้อมูลไปเพิ่มในตาราง Student โดยพิมพ์คำสั่งดังนี้

```

1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:orientation="vertical"
8      tools:context=".InsertActivity">
9      <TextView
10         android:layout_width="match_parent"
11         android:layout_height="wrap_content"
12         android:text="Add Student"
13         android:textStyle="bold"
14         android:textAlignment="center"
15         android:textSize="25sp"/>
16
17     <EditText
18         android:id="@+id/edt_id"
19         android:layout_width="match_parent"
20         android:layout_height="wrap_content"
21         android:inputType="text"
22         android:textSize="20sp"
23         android:hint="Input ID"/>
24
25     <EditText
26         android:id="@+id/edt_name"
27         android:layout_width="match_parent"
28         android:layout_height="wrap_content"
29         android:inputType="text"
30         android:textSize="20sp"
31         android:hint="Input Name"/>
32
33     <EditText
34         android:id="@+id/edt_age"
35         android:layout_width="match_parent"
36         android:layout_height="wrap_content"
37         android:inputType="number"
38         android:textSize="20sp"
39         android:hint="Input Age"/>
40     <Button
41         android:id="@+id/btnAdd"
42         android:layout_width="match_parent"
43         android:layout_height="wrap_content"
44         android:layout_marginTop="10dp"
45         android:background="#53deed"
46         android:text="Add"
47         android:textSize="20sp"
48         android:onClick="addStudent" />

```

```

49 <Button
50     android:id="@+id/btnCancel"
51     android:layout_width="match_parent"
52     android:layout_height="wrap_content"
53     android:layout_marginTop="10dp"
54     android:background="#FC3C71"
55     android:text="Reset"
56     android:textSize="20sp"
57     android:onClick="resetStudent"/>
58 </LinearLayout>

```

2. ถัดมาพิมพ์คำสั่งในไฟล์ InsertActivity ดังนี้

```

14 class InsertActivity : AppCompatActivity() {
15
16     override fun onCreate(savedInstanceState: Bundle?) {
17         super.onCreate(savedInstanceState)
18         setContentView(R.layout.activity_insert)
19     }
20     fun addStudent(v: View){
21
22         val api : StudentAPI = Retrofit.Builder()
23             .baseUrl( baseUrl: "http://10.0.2.2:3000/")
24             .addConverterFactory(GsonConverterFactory.create())
25             .build()
26             .create(StudentAPI ::class.java)
27
28         api.insertStd(
29             edt_id.text.toString(),
30             edt_name.text.toString(),
31             edt_age.text.toString().toInt()).enqueue(object : Callback<Student> {
32
33             override fun onResponse(call: Call<Student>, response: retrofit2.Response<Student>) {
34
35                 if (response.isSuccessful()) {
36
37                     Toast.makeText(applicationContext, text: "Successfully Inserted", Toast.LENGTH_SHORT).show()
38                     finish()
39                     // Log.i("Test", "post registration to API" + response.body()!!.toString())
40
41                 }else{
42                     Toast.makeText(applicationContext, text: "Error ", Toast.LENGTH_SHORT).show()
43                 }
44             }
45
46             override fun onFailure(call: Call<Student>, t: Throwable) {
47                 Toast.makeText(applicationContext, text: "Error onFailure " + t.message, Toast.LENGTH_LONG).show()
48             }
49         })
50     }
51
52     fun resetStudent(v: View){
53
54         edt_id.getText().clear()
55         edt_name.getText().clear()
56         edt_age.getText().clear()
57     }
58
59 }

```

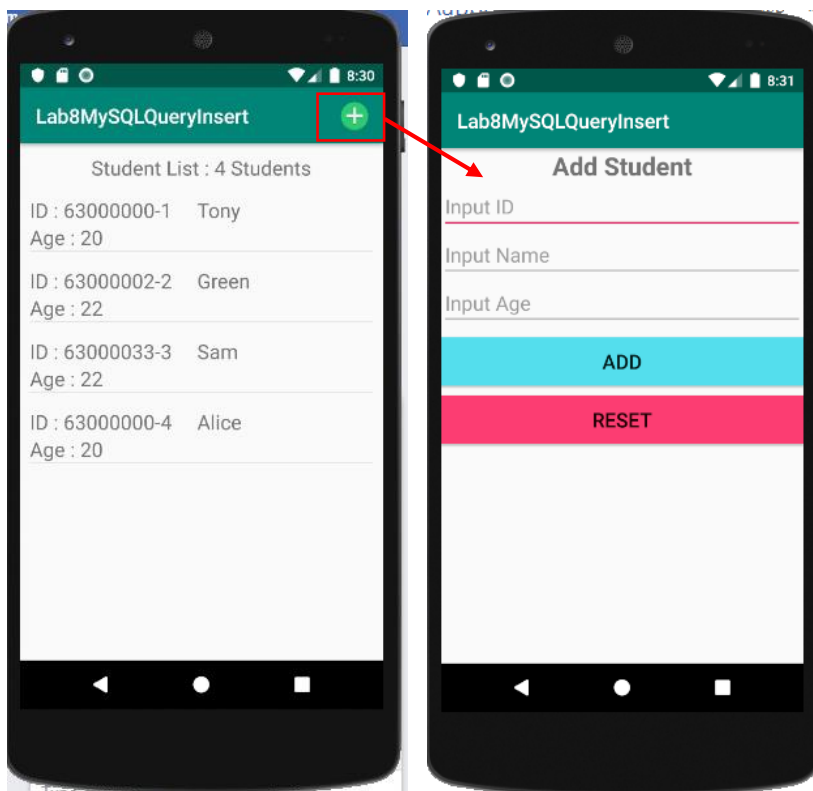
3. จากนั้นให้ไปที่ไฟล์ MainActivity เพื่อเขียนคำสั่งเพิ่มในส่วนของการเพิ่ม Option Menu ในการเรียกใช้งาน InsertActivity

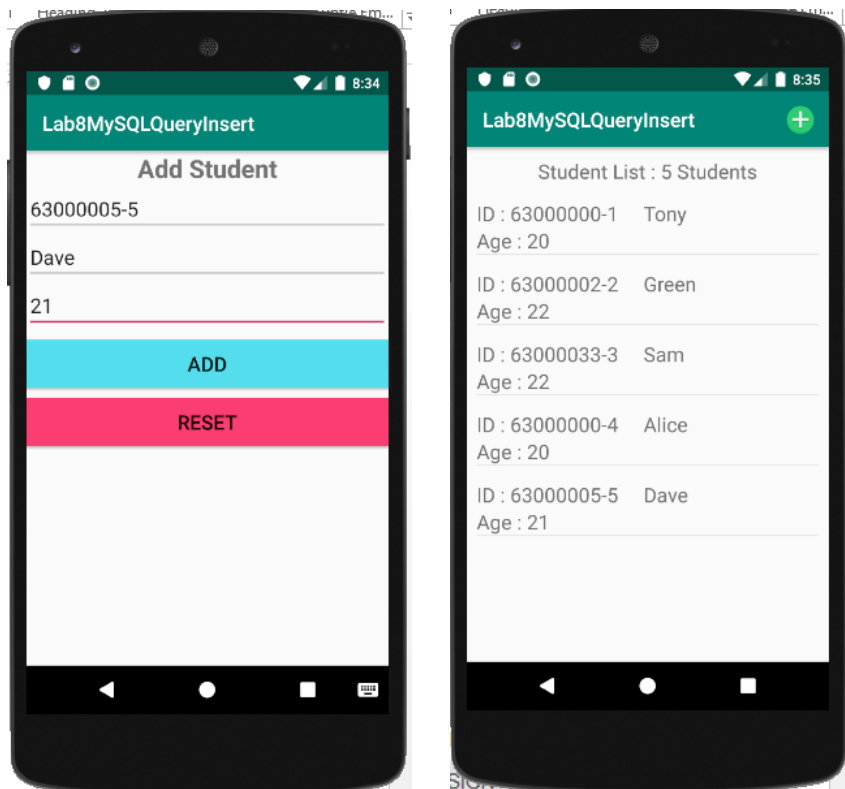
```

39
40 override fun onResume() {
41     super.onResume()
42     callStudentData()
43 }
44
45 override fun onCreateOptionsMenu(menu: Menu): Boolean {
46     menuInflater.inflate(R.menu.option_menu, menu)
47     return super.onCreateOptionsMenu(menu)
48 }
49
50 override fun onOptionsItemSelected(item: MenuItem): Boolean {
51     val id: Int = item.itemId
52     when (id) {
53         R.id.item1 -> {
54             val intent = Intent(packageContext: this@MainActivity, InsertActivity::class.java)
55             startActivity(intent)
56             return true
57         }
58         else -> return super.onOptionsItemSelected(item)
59     }
60 }
61
62

```

จากนั้นให้ Run โปรแกรมจะแสดงผล ดังนี้





หลังจากนั้นให้เพิ่มข้อมูลในแอปพลิเคชัน แล้วให้ตรวจสอบค่าที่อยู่ใน Database ว่ามีข้อมูลเพิ่มหรือไม่

No	std_id	std_name	std_age
1	63000000-1	Tony	20
2	63000002-2	Green	22
3	63000033-3	Sam	22
4	63000000-4	Alice	20
5	63000005-5	Dave	21

Assignment 8

ให้นักศึกษาสร้างแอปพลิเคชันที่แสดงข้อมูลของรหัสนักศึกษาโดยสร้างฐานข้อมูลโดยมีข้อมูลดังนี้
 emp_id Primary Key (Auto Increment), emp_name , emp_gende , emp_email และ emp_alary
 โดยทำฟอร์มกรอกข้อมูลดังภาพด้านล่างและแสดงที่ RecyclerView โดยหน้าจอตั้งภาพด้านล่าง และเมื่อกดปุ่ม
 Add Employee จะแสดง หน้าจออีกอันเพื่อรับข้อมูล ในการเพิ่มข้อมูลพนักงาน แล้วเมื่อกรอกข้อมูลพนักงาน
 แล้วกดที่ปุ่ม Add ก็จะทำให้การแสดงผลข้อมูลของพนักงานเพิ่มที่ Recycleview ในหน้าแรก

