

Modul 2
Pemrograman Berbasis Platform
Android Kotlin

Introduction View Binding, Data Binding, MVVM and Shared Preference

Data binding and Data Persistence part 1



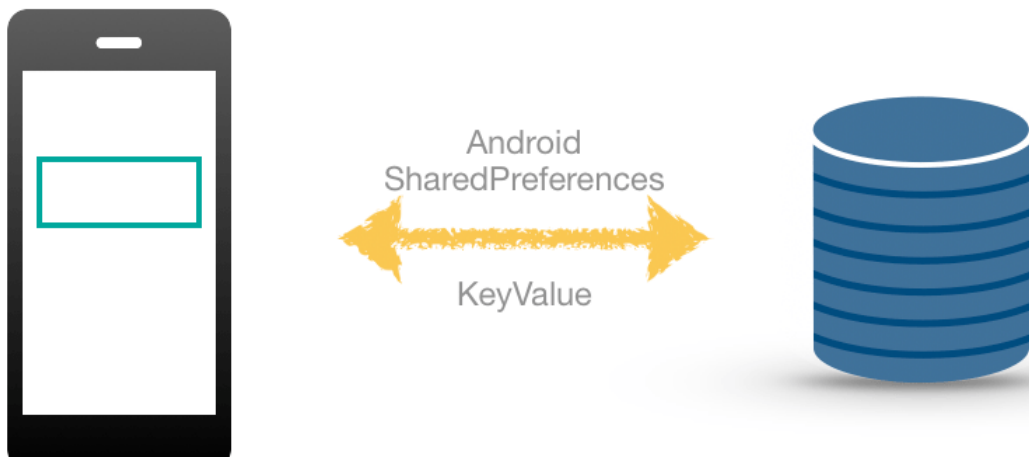
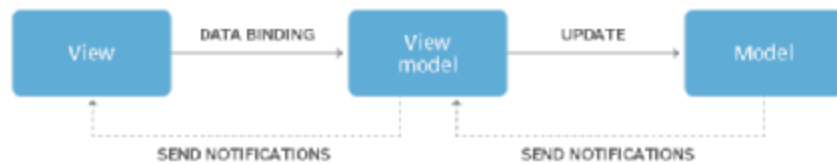
Oleh:

Chris YPT – 180709999

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ATMA JAYA YOGYAKARTA
2022

View Binding vs Data Binding

Model-View-ViewModel



Pemrograman Berbasis Platform

Android Kotlin

Tujuan modul ini adalah :

1. Mengenalkan konsep *view binding* serta penggunaan
2. Mengenalkan konsep *data binding* serta penggunaan
3. Perbedaan antara *view binding* dengan *data binding* dalam eksekusi aplikasi
4. Mengenalkan konsep *shared preference* secara sederhana
5. Menjelaskan konsep MVVM (Model – View – ViewModel) dalam proses pengembangan app

A. VIEW BINDING

Jika kita merujuk kepada dokumentasi dari *Android Developer*, view binding adalah sebuah fitur yang mana kalian dimudahkan untuk menuliskan kode yang berinteraksi dengan tampilan. View binding akan membentuk *class binding* yang mana setiap file dari XML yang ada akan berinteraksi dengan variable pada class yang kalian atur. Karena kemudahan ini, kalian langsung eksekusi kepada XML yang bereferensi langsung yang ada ID di tata letak tersebut. View binding bisa dikatakan pengganti dari *findViewById*.

Jadi, saat kita menggunakan View Binding, semua layout yang kita miliki otomatis akan di-generate oleh Android Studio menjadi file binding. Di dalam file binding, semua View yang ada di dalam layout tersebut dibuatkan variable *findViewById*. Itulah mengapa kita dapat mengakses semua View yang ada di layout tersebut.

1. Guided I

Kali ini kita akan sama-sama belajar membuat project sederhana dari view binding pakai *recyclerview*.

- a. Kalian buat sebuah project dengan ketentuan seperti berikut :
- Activity = Empty Activity
 - Name project = view_binding_5digitnpm
 - Save location = (bebas sesuaikan dengan disk kalian)
 - Minimum SDK = API 26 (Oreo Android 8.0)
- b. Setelah Gradle kalian sudah selesai di build dan siap ngoding, kalian pergi ke build.gradle (*Module:....*) kemudian kalian tambahkan code seperti ada pada gambar berikut. Kemudian kalian **sync now**

A screenshot of an Android Studio build.gradle file. The code is as follows:

```
viewBinding {  
    enabled = true  
}  
  
dependencies {  
  
    implementation 'androidx.core:core-ktx:1.7.0'  
    implementation 'androidx.appcompat:appcompat:1.5.0'  
    //implemetasi buat recyclerview  
    implementation 'androidx.recyclerview:recyclerview:1.2.1'  
}
```

Two red hand-drawn circles highlight specific parts of the code. The first circle is around the `viewBinding { enabled = true }` block. The second circle is around the `implementation 'androidx.recyclerview:recyclerview:1.2.1'` line, which is preceded by a comment `//implemetasi buat recyclerview`. The word 'implemetasi' in the comment is misspelled.

Gambar 1 Penambahan code untuk view binding

- c. Setelah melakukan import library untuk viewBinding dan recyclerview, kalian pergi ke Res -> layout -> activity_main. Kemudian kalian tambahkan code seperti berikut : (*untuk tools:listItem kalian jadikan comment dulu saja.*).

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout 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=".MainActivity">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/taskRv"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:listitem="@layout/recyclerview_item"
        app:layoutManager="androidx.recyclerview.widget.LinearLayoutManager"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:viewBindingIgnore="true"/>

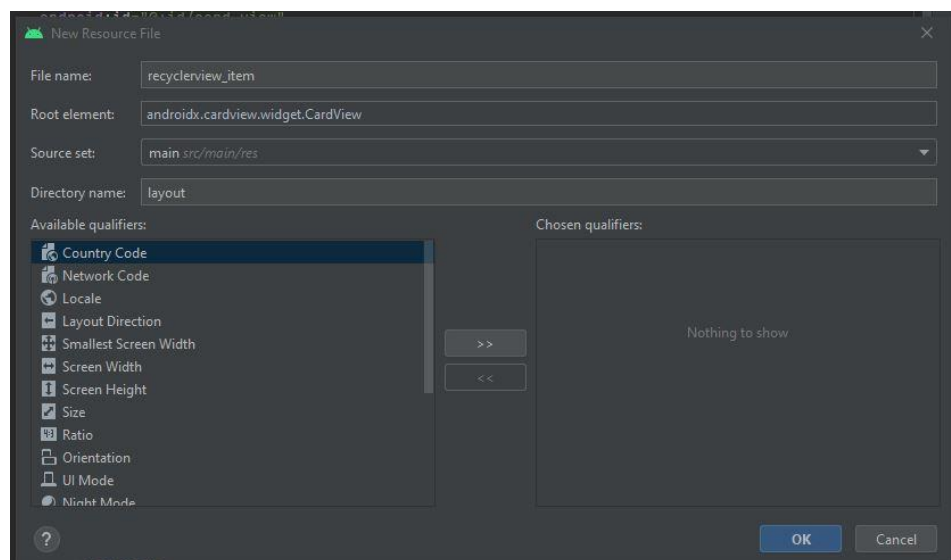
</androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 2 kode untuk activity_main layout

d. disini kita akan membuat sebuah cardview yang mana diisi oleh beberapa kata-kata dan image sederhana yang statis. Untuk itu kalian diwajibkan untuk membuat cardview. Caranya adalah kalian pergi ke layout pada res, kemudian klik kanan pilih New-> Layout Resource File lalu akan muncul kotak dialog New Resource File. Lalu kalian isikan seperti berikut :

- File Name = recyclerview_item
- Root Elemen = androidx.cardview.widget.CardView

Lalu klik OK.



Gambar 3 proses pembuatan recyclerview_item

- e. Kemudian kalian buka recyclerview_item.xml kalian lalu ikutin code seperti berikut :

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/card_view"
    android:layout_margin="5dp"
    app:cardBackgroundColor="#81C784"
    app:cardCornerRadius="12dp"
    app:cardElevation="3dp"
    app:contentPadding="4dp">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:id="@+id/relativeLayout"
        android:layout_width="match_parent"
        android:layout_height="181dp"
        android:padding="16dp">

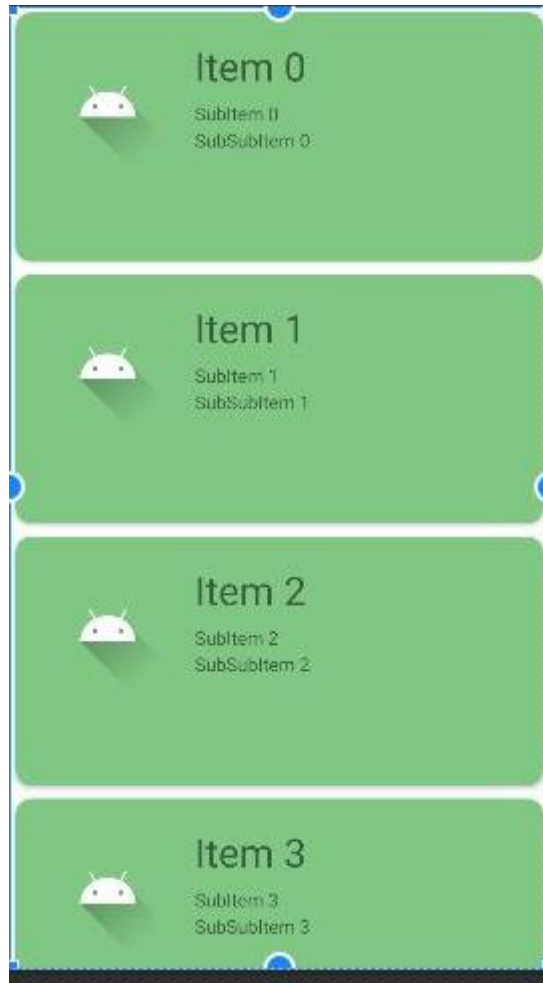
        <ImageView
            android:id="@+id/item_image"
            android:layout_width="100dp"
            android:layout_height="100dp"
            android:src="@drawable/ic_launcher_foreground"
            app:layout_constraintLeft_toLeftOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            tools:ignore="ImageContrastCheck"
            android:contentDescription="TODO" />

        <TextView
            android:id="@+id/chapter"
            android:layout_width="236dp"
            android:layout_height="39dp"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintLeft_toRightOf="@id/item_image"
            android:layout_marginStart="16dp"
            android:textSize="30sp"/>

        <TextView
            android:id="@+id/itemTitle"
            android:layout_width="236dp"
            android:layout_height="16dp"
            android:layout_marginStart="16dp"
            android:layout_marginTop="8dp"
            app:layout_constraintTop_toBottomOf="@id/chapter"
            app:layout_constraintLeft_toRightOf="@id/item_image"/>

        <TextView
            android:id="@+id/itemDetail"
            android:layout_width="236dp"
            android:layout_height="32dp"
            android:layout_marginStart="16dp"
            android:layout_marginTop="4dp"
            android:ellipsize="none"
            android:scrollHorizontally="true"
            app:layout_constraintLeft_toRightOf="@id/item_image"
            app:layout_constraintTop_toBottomOf="@id/itemTitle" />
    </androidx.constraintlayout.widget.ConstraintLayout>
</androidx.cardview.widget.CardView>
```

- f. Lalu Kembali ke `activity_main` kemudian kalian un-comment `tools:listitem` tersebut maka hasilnya akan seperti berikut :



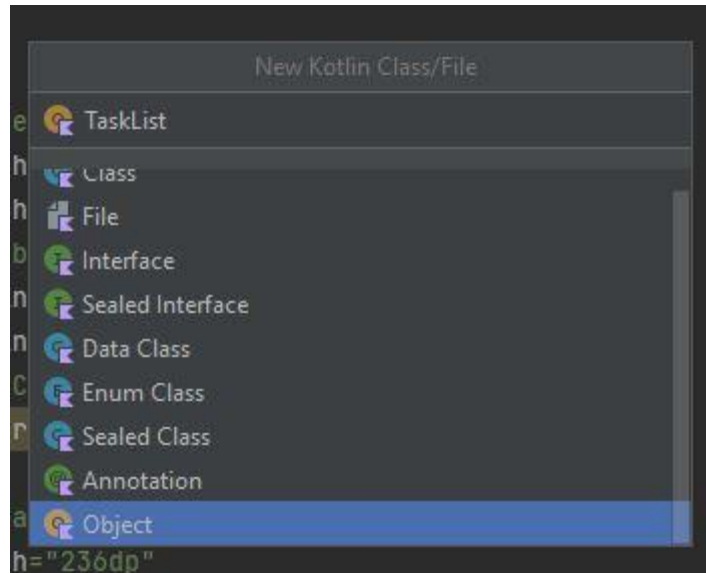
Gambar 4 hasil akhir untuk layout recycler view

- g. Sekarang kita ke backend untuk nulis coding model-viewmodel yang akan kita buat. Untuk itu kalian pergi ke `java->directory` nama project kalian. Klik kanan pada directory project kalian lalu pilih `New-> Kotlin Class/File`. Lalu buatlah sebuah class kotlin dengan nama `Task.kt`. lalu kalian isikan dalam class tersebut pada code berikut :

```
data class Task (val chapter:String, val title:String, val details:String)
```

Gambar 5 Data class Task.kt

- h. Kemudian kalian buat sebuah file object dengan nama object.kt. caranya adalah klik kanan pada directory project kalian lalu pilih New -> Kotlin Class/File. Kemudian pilih object dan buat Nama class object jadi TaskList.



Gambar 6 Object Class kotlin

- i. Lalu kalian bisa ikutin code seperti berikut : (Isinya bebas kalian bisa isi kata-katanya sendiri boleh. **Kalian hanya tinggal tanda petik dua langsung isi saja**)

```
object TaskList {  
    val taskList = listOf<Task> (  
        Task( chapter: "Chapter 1", title: "Bangun Tidur",  
            details: "Kebiasaan sehari-hari untuk membuat hari makin indah"),  
        Task( chapter: "Chapter 2", title: "Nguli Genshin Impact", details: "Mencari Primo buat Gacha Mama Yaemiko tercinta / Keding Wangi"),  
        Task( chapter: "Chapter 3", title: "Ngopi", details: "Habis capek Nguli, kan enak ngopi ya kan"),  
        Task( chapter: "Chapter 4", title: "dimana mama Ei?", details: "Buat Adu Mekanik Showcase Ayaka Pyro"),  
        Task( chapter: "chapter 5", title: "Upload ke Tiktok",  
            details: "Buat Pamer ke view Showcase Ayaka Pyro C6 + Bennet C6")  
    )  
}
```

Gambar 7 Isi Object class TaskList

- j. Kemudian kalian kembali membuat sebuah kelas lagi Bernama ActivityAdapter.kt cara sama seperti membuat class Task.kt tapi dengan nama filenya adalah MainAdapter.kt. lalu kalian isikan kode seperti berikut :

```
class MainAdapter(val taskList: List<Task>):RecyclerView.Adapter<MainAdapter.MainViewHolder>() {
    inner class MainViewHolder (val itemBinding: RecyclerViewItemBinding)
        :RecyclerView.ViewHolder(itemBinding.root) {
        fun bindItem(task:Task){
            itemBinding.chapter.text = task.chapter
            itemBinding.itemTitle.text = task.title
            itemBinding.itemDetail.text = task.details
        }
    }

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): MainViewHolder {
        return MainViewHolder(RecyclerViewItemBinding.inflate(LayoutInflater.from(parent.context),parent, attachToParent: false))
    }

    override fun onBindViewHolder(holder: MainViewHolder, position: Int) {
        val task = taskList[position]
        holder.bindItem(task)
    }

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

Gambar 8Adapter Main untuk controlling View-ViewModel

- k. Terakhir kalian pergi ke MainActivity.kt dan isikan kode seperti berikut :

```
class MainActivity : AppCompatActivity() {

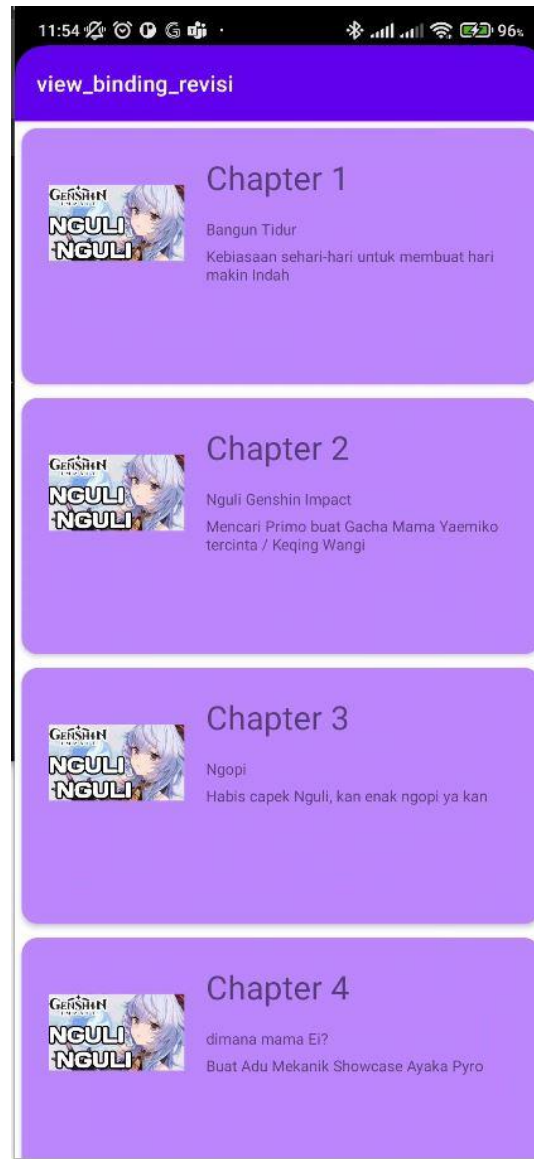
    var binding: ActivityMainBinding? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        // setContentView(R.layout.activity_main)

        binding = ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)
        val adapter = MainAdapter(TaskList.taskList)
        binding?.taskRv?.adapter = adapter
    }

    override fun onDestroy() {
        super.onDestroy()
        binding = null
    }
}
```

1. Silakan kalian running maka hasilnya seperti ini :



Gambar 9 hasil akhir

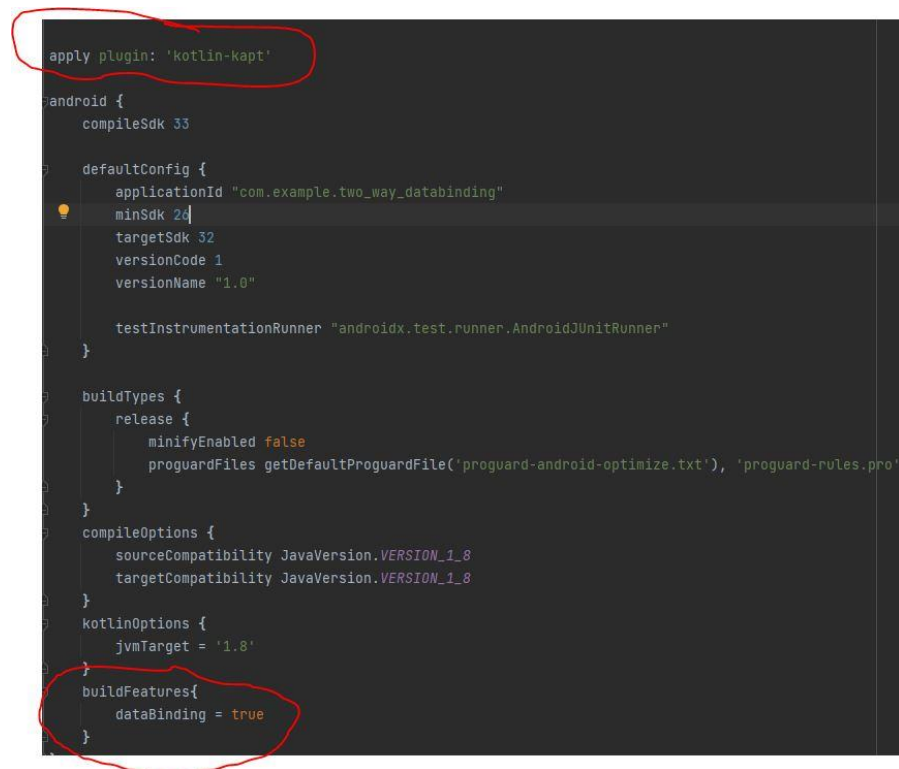
B. DATA BINDING

Menurut dari Android Developer, Data Binding adalah salah satu bagian dari support library yang fungsinya untuk memudahkan kita dalam menyambungkan/mengikat sebuah data ke bagian UInya, sehingga kita tidak perlu melakukan set data secara manual melalui bagian kode programnya / penulisan secara deklaratif.

Guided II

Kali kita akan belajar secara sederhana membuat sebuah app Data binding dengan Live Data.

- a. Kalian buat sebuah project dengan ketentuan seperti berikut :
 - Activity = Empty Activity
 - Name project = data_binding_5digitnpm
 - Save location = (bebas sesuaikan dengan disk kalian)
 - Minimum SDK = API 26 (Oreo Android 8.0)
- b. Selesai Build gradle kalian, kalian pergi ke Build.gradle (*module:....*) lalu kalian tambahkan kode seperti ini :



```
apply plugin: 'kotlin-kapt'

android {
    compileSdk 33

    defaultConfig {
        applicationId "com.example.two_way_databinding"
        minSdk 24
        targetSdk 32
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
    kotlinOptions {
        jvmTarget = '1.8'
    }
    buildFeatures{
        dataBinding = true
    }
}
```

lalu kalian **Sync Now**.

- c. Selesai dari gradlenya, kalian pergi ke Activity_main.xml (layout) kalian.
Kemudian kalian kode seperti berikut :

```
<?xml version="1.0" encoding="utf-8"?>
<layout 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">
    <data>
        <variable
            name="viewmodel"
            type="com.example.two_way_databinding.viewmodel.MainViewModel" />
    </data>

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:context=".MainActivity">

        <TextView
            android:id="@+id/textView"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@{viewmodel.currentRandomFruitName}"
            android:textAppearance="@style/TextAppearance.AppCompat.Display1"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintLeft_toLeftOf="parent"
            app:layout_constraintRight_toRightOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent"
            app:layout_constraintVertical_bias="0.12"
            tools:text="Some random fruit"/>
    </androidx.constraintlayout.widget.ConstraintLayout>
</layout>
```

<Button

```
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:layout_marginBottom="8dp"
    android:backgroundTint="@color/baby_pink"
    android:onClick="@{() -> viewModel.onChangeRandomFruitClick()}"
    android:text="Change Fruit"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@id/textView"
    app:layout_constraintVertical_bias="0.0"/>
```

<EditText

```
    android:id="@+id/editText"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginBottom="8dp"
    android:hint="Enter a fruit name"
    android:text="@={viewModel.editTextContent}"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.491"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@id/button"
    app:layout_constraintVertical_bias="0.19"
    app:layout_constraintWidth_percent="0.8"/>
```



```

<Button
    android:id="@+id/button2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginTop="8dp"
    android:layout_marginBottom="8dp"
    android:backgroundTint="@color/teal_700"
    android:text="Display EditText content"
    android:onClick="@{() -> viewModel.onDisplayEditTextContentClick()}"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@id/editText"
    app:layout_constraintVertical_bias="0.0"/>

```

```

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginBottom="8dp"
    android:text="@{viewModel.displayedEditTextContent}"
    android:textAppearance="@style/TextAppearance.AppCompat.Display1"
    app:layout_constraintBottom_toTopOf="@id/button3"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@id/button2"
    app:layout_constraintVertical_bias="0.501"
    tools:text="Content od Edit Text"
/>

```

```

<Button
    android:id="@+id/button3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:layout_marginEnd="8dp"
    android:layout_marginBottom="8dp"
    style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"
    android:text="select random fruit for EditText"
    android:onClick="@{() -> viewModel.onSelectRandomEditTextFruit()}"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.511"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@id/button2"
    app:layout_constraintVertical_bias="0.475"
    tools:ignore="HardcodedText" />

</androidx.constraintlayout.widget.ConstraintLayout>

</layout>

```

- d. Kemudian kalian buat sebuah file class object dengan nama FakeRepository.kt dengan cara pergi ke directory project kalian lalu buat New -> Kotlin Class/File lalu pilih object class dengan nama FakeRepository.kt. kemudian kalian isikan kode seperti berikut :

```

object FakeRepository {
    private val fruitNames: List<String> = listOf(
        "Apel", "mangga", "Pisang", "Anggur", "Stroberi",
        "Jambu", "Raspberi", "Bambu", "Kiwi", "Pir"
    )

    private val _currentRandomFruitName = MutableLiveData<String>()
    val currentRandomFruitName: LiveData<String>
        get() = _currentRandomFruitName

    init {
        _currentRandomFruitName.value = fruitNames.first()
    }

    fun getRandomfruitName(): String {
        val random = Random.nextInt(fruitNames.size)
        return fruitNames[random]
    }

    fun changeCurrentRandomFruitName() {
        _currentRandomFruitName.value = getRandomfruitName()
    }
}

```

- e. Saat kita akan mengontrol bagian View dengan ViewModel untuk komunikasi dengan UI komponen tersebut. Oleh karena itu, kalian buat sebuah class baru dengan New-> Kotlin Class/File lalu pilih class dengan nama MainViewModel.kt .
- f. Setelah itu, kalian buka MainViewModel.kt lalu ikutin kode seperti berikut :


```

class MainViewModel: ViewModel() {
    val currentRandomFruitName: LiveData<String>
        get() = FakeRepository.currentRandomFruitName

    fun onChangeRandomFruitClick() = FakeRepository.changeCurrentRandomFruitName()

    val editTextContent = MutableLiveData<String>()

    private val _displayedEditTextContent = MutableLiveData<String>()
    val displayedEditTextContent: LiveData<String>
        get() = _displayedEditTextContent

    fun onDisplayEditTextContentClick(){
        _displayedEditTextContent.value = editTextContent.value
    }

    fun onSelectRandomEditTextFruit() {
        editTextContent.value = FakeRepository.getRandomfruitName()
    }
}

```

g. Terakhir, kalian buka MainActivity.kt kalian lalu ketik sebagai berikut :

```

class MainActivity : AppCompatActivity() {

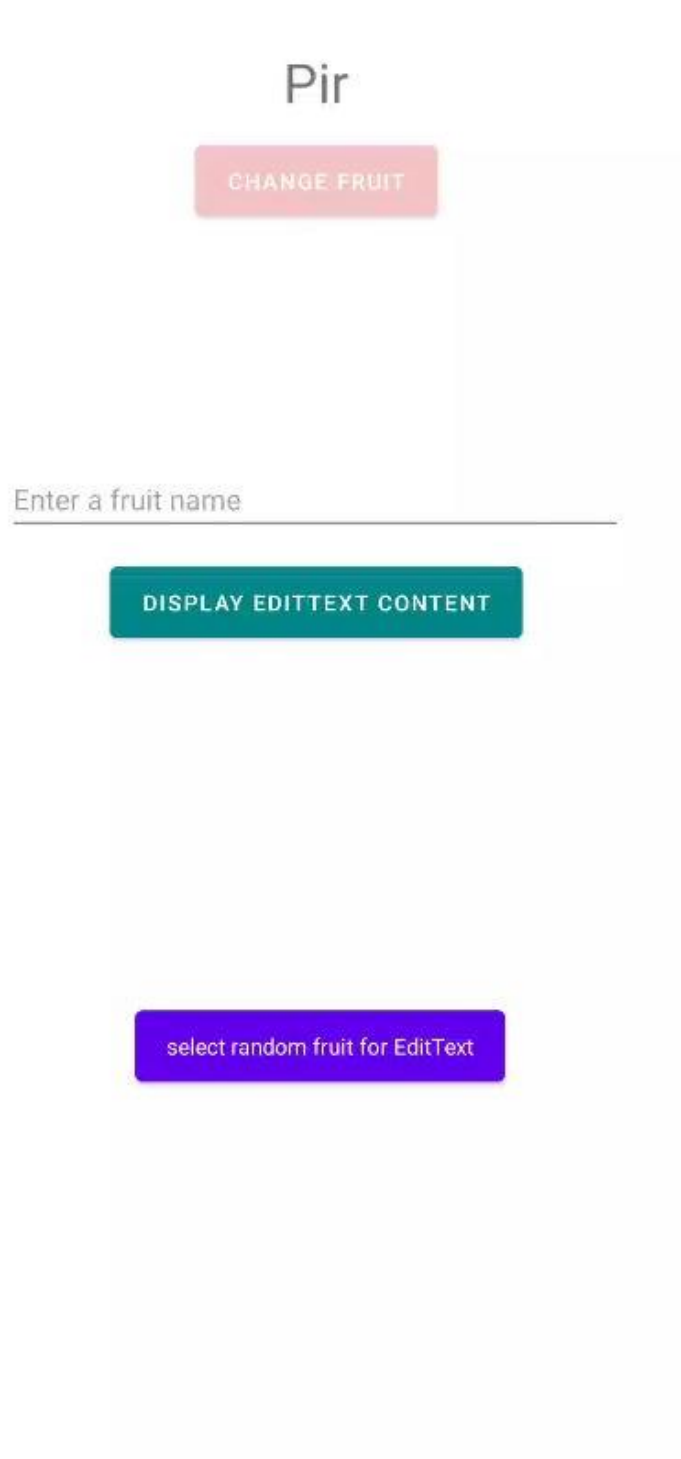
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val mainViewModel = ViewModelProvider( owner: this).get(MainViewModel::class.java)
        DataBindingUtil.setContentView<ActivityMainBinding>(
            activity: this,R.layout.activity_main
        ).apply { this: ActivityMainBinding!
            this.lifecycleOwner = this@MainActivity
            this.viewmodel = mainViewModel
        }

        mainViewModel.editTextContent.observe( owner: this) { it: String!
            Toast.makeText( context: this, it, Toast.LENGTH_SHORT).show()
        }
    }
}

```

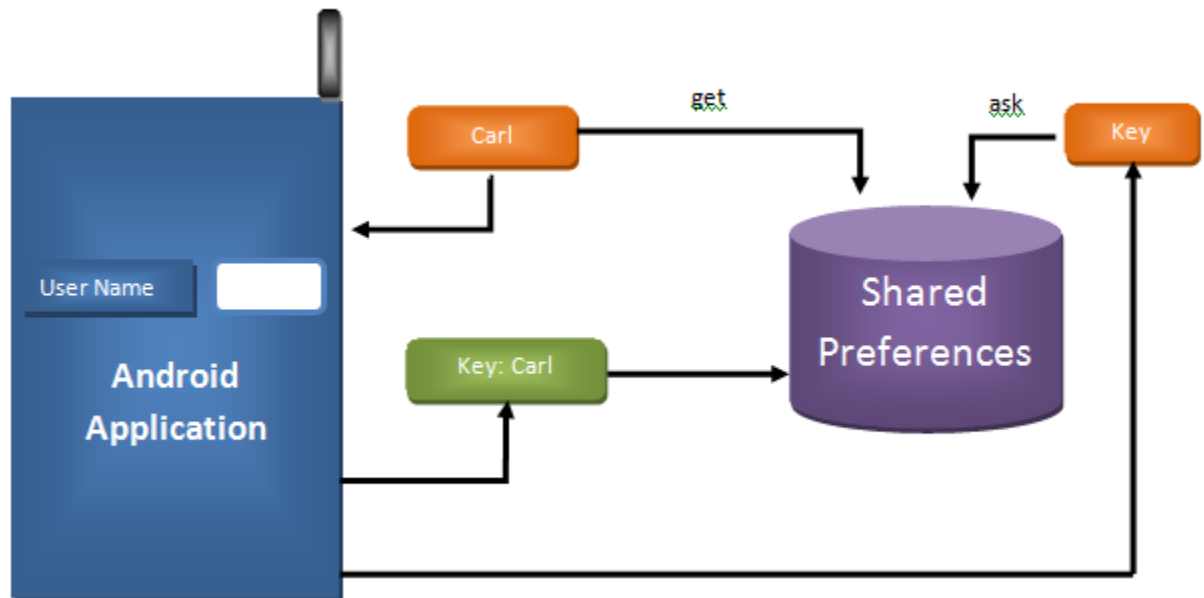
h. Silakan Running hasilnya sebagai berikut :



The screenshot displays an Android application interface. At the top, the word "Pir" is shown in a large, dark font. Below it is a pink button labeled "CHANGE FRUIT". Further down is a text input field with the placeholder text "Enter a fruit name". Below the input field is a teal button labeled "DISPLAY EDITTEXT CONTENT". At the bottom of the visible area is a purple button labeled "select random fruit for EditText". The background of the application is a light gray with a subtle grid pattern.

C. SHARED PREFERENCE

Shared Preference adalah sebuah media penyimpanan yang hanya bisa menyimpan data sederhana berupa data primitf (seperti *integer*, *float*, *Long*, *String*, dan *Boolean*) dalam file internal yang ada di aplikasi dalam bentuk **key-value**. Prinsip kerja seperti ini : dalam



sebuah field yang dimasukkan data bertipe integer misalkan. Field yang memiliki key-value berupa id akan diteruskan kepada key yang diinisialisasi pada class MainActivity.kt untuk bertanya apakah key id field sudah bisa dimasukkan datanya. Dengan memakai shared Preference, akan dicari key yang sama dengan id di field tersebut. Jika sama, maka id field akan mendapatkan data yang sudah disimpan dalam shared preference dengan get ke id xml tersebut.

GUIDED III.

Kali kita akan mencoba membuat app sederhana dengan adanya Save, retrieve, dan delete data dengan Share Preference.

a. Kalian buat sebuah project dengan ketentuan seperti berikut :

- Activity = Empty Activity
- Name project = shareP_5digitnpm
- Save location = (bebas sesuaikan dengan disk kalian)
- Minimum SDK = API 26 (Oreo Android 8.0)

- b. Kita akan membuat 1 activity saja dengan memakai Activity_main.xml saja.
Silakan kalian kerjakan code seperti berikut :

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="8dp"
    tools:context=".MainActivity">
    <Button
        style="@style/Widget.Material3.Button.ElevatedButton"
        android:id="@+id/btnSave"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_centerInParent="true"
        android:onClick="saveData"
        android:text="Save"
        android:textColor="@color/black"
        android:textStyle="bold"
        android:backgroundTint="@color/green"
        android:textSize="15dp"
        tools:ignore="OnClick" />
    <Button
        style="@style/Widget.Material3.Button.ElevatedButton"
        android:id="@+id/btnRetrieve"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"
        android:layout_centerHorizontal="true"
        android:onClick="readData"
        android:text="Read"
        android:textStyle="bold"
        android:textColor="@color/white"
        android:backgroundTint="@color/blue"
        android:textSize="15dp"
        tools:ignore="OnClick" />
    <Button
        style="@style/Widget.Material3.Button.ElevatedButton"
        android:id="@+id/btnClear"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentEnd="true"
        android:layout_centerInParent="true"
        android:onClick="clearData"
        android:text="Clear"
        android:textStyle="bold"
        android:textColor="@color/white"
        android:backgroundTint="@color/red"
        android:textSize="15dp"
        tools:ignore="OnClick" />
    <EditText
        android:id="@+id/etEmail"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
```

```

        android:layout_below="@+id/etName"
        android:layout_alignParentEnd="true"
        android:layout_marginTop="10dp"
        android:ems="10"
        android:hint="Email"
        android:inputType="textEmailAddress" />
    <EditText
        android:id="@+id/etName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignStart="@+id/etEmail"
        android:layout_marginTop="40dp"
        android:ems="10"
        android:hint="Name"
        android:inputType="text" />
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="400dp"
        android:layout_below="@+id/btnSave"
        android:layout_marginTop="10dp"
        android:orientation="vertical"
        android:padding="8dp">
        <TextView
            android:id="@+id/textViewName"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="20dp"
            android:textColor="@android:color/holo_blue_light"
            android:textSize="24sp"
            android:textStyle="bold" />
        <TextView
            android:id="@+id/textViewEmail"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="10dp"
            android:textColor="@android:color/holo_blue_light"
            android:textSize="24sp"
            android:textStyle="bold" />
    </LinearLayout>
</RelativeLayout>

```

- c. Kemudian, kalian pergi ke MainActivity.kt lalu kalian kode seperti berikut :

```

@Suppress("NULLABILITY_MISMATCH_BASED_ON_JAVA_ANNOTATIONS")
class MainActivity : AppCompatActivity() {

    var editTextName: EditText? = null
    var editTextEmail: EditText? = null
    lateinit var textViewName: TextView
    lateinit var textViewEmail: TextView
    private val myPreference = "myPref"
    private val name = "nameKey"
    private val email = "emailKey"
    var sharedPreferences: SharedPreferences? = null

```

```

        override fun onCreate(savedInstanceState: Bundle?) {
            super.onCreate(savedInstanceState)
            setContentView(R.layout.activity_main)
            title = "KotlinApp"
            editTextEmail = findViewById(R.id.etEmail)
            editTextName = findViewById(R.id.etName)
            sharedPreferences = getSharedPreferences(myPreference,
Context.MODE_PRIVATE)
            if (sharedPreferences!!.contains(name)) {

editTextName?.setText(sharedPreferences!!.getString(name, ""))
            }
            if (sharedPreferences!!.contains(email)) {

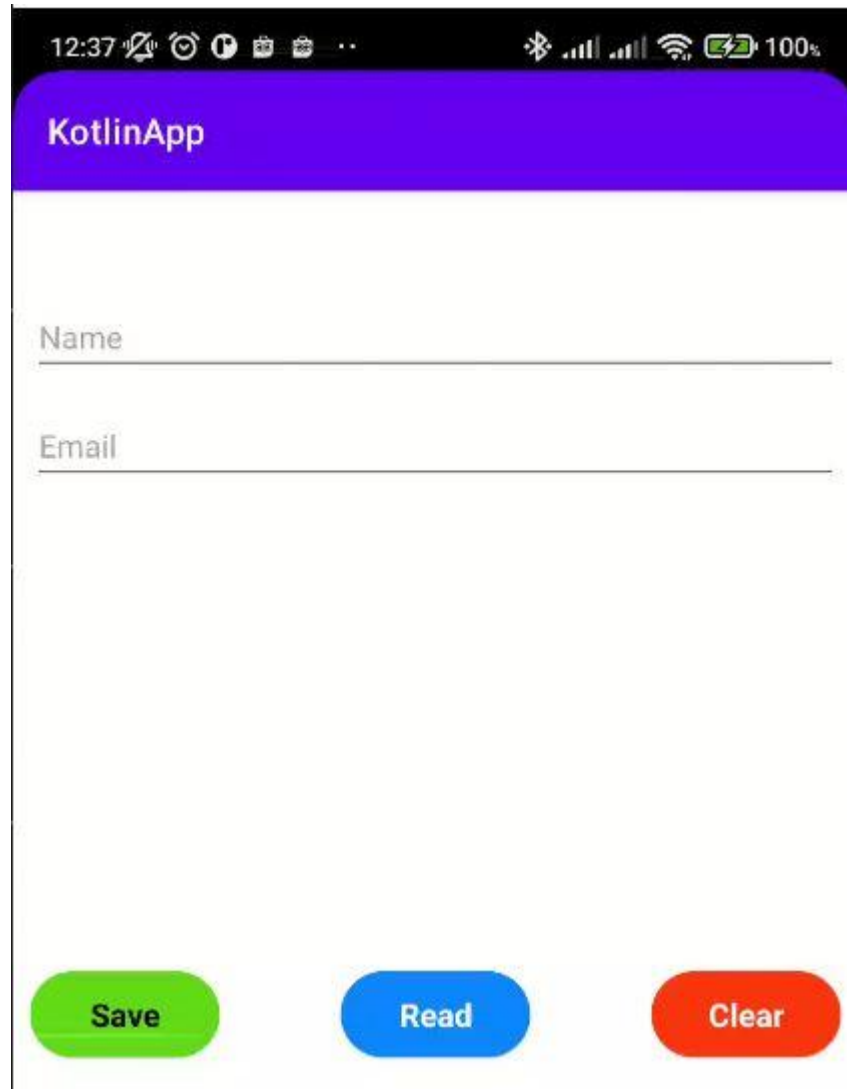
editTextName?.setText(sharedPreferences!!.getString(email, ""))
            }
        }
        fun readData(view: View) {
            textViewEmail = findViewById(R.id.textViewEmail)
            textViewName = findViewById(R.id.textViewName)
            var strName: String =
editTextName?.text.toString().trim()
            var strEmail: String =
editTextEmail?.text.toString().trim()
            strName = sharedPreferences!!.getString(name, "")!!
            strEmail = sharedPreferences!!.getString(email, "")!!
            sharedPreferences = getSharedPreferences(myPreference,
Context.MODE_PRIVATE)
            if (sharedPreferences!!.contains(name)) {
                textViewName.text = strName
            }
            if (sharedPreferences!!.contains(email)) {
                textViewEmail.text = strEmail
            }
            Toast.makeText(baseContext, "Data retrieved",
Toast.LENGTH_SHORT).show()
        }
        fun saveData(view: View) {
            val strName: String =
editTextName?.text.toString().trim()
            val strEmail: String =
editTextEmail?.text.toString().trim()
            val editor: SharedPreferences.Editor =
sharedPreferences!!.edit()
            editor.putString(name, strName)
            editor.putString(email, strEmail)
            editor.apply()
            Toast.makeText(baseContext, "Saved",
Toast.LENGTH_SHORT).show()
        }

        fun clearData(view: View) {
            editTextName!!.text.clear()
            editTextEmail!!.text.clear()
            textViewName.text = ""
            textViewEmail.text = ""
        }
    }
}

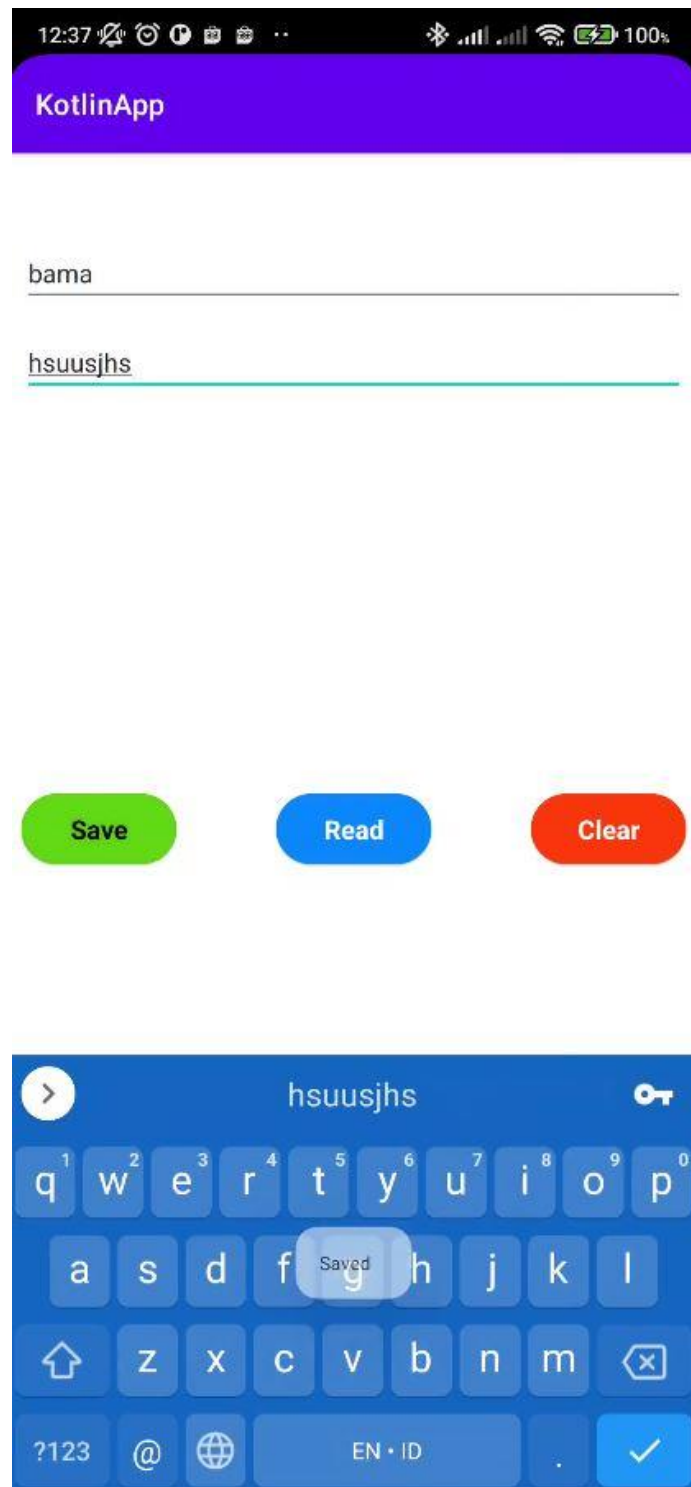
```

```
Toast.makeText(baseContext, "Cleared data",  
Toast.LENGTH_SHORT).show()  
}  
}
```

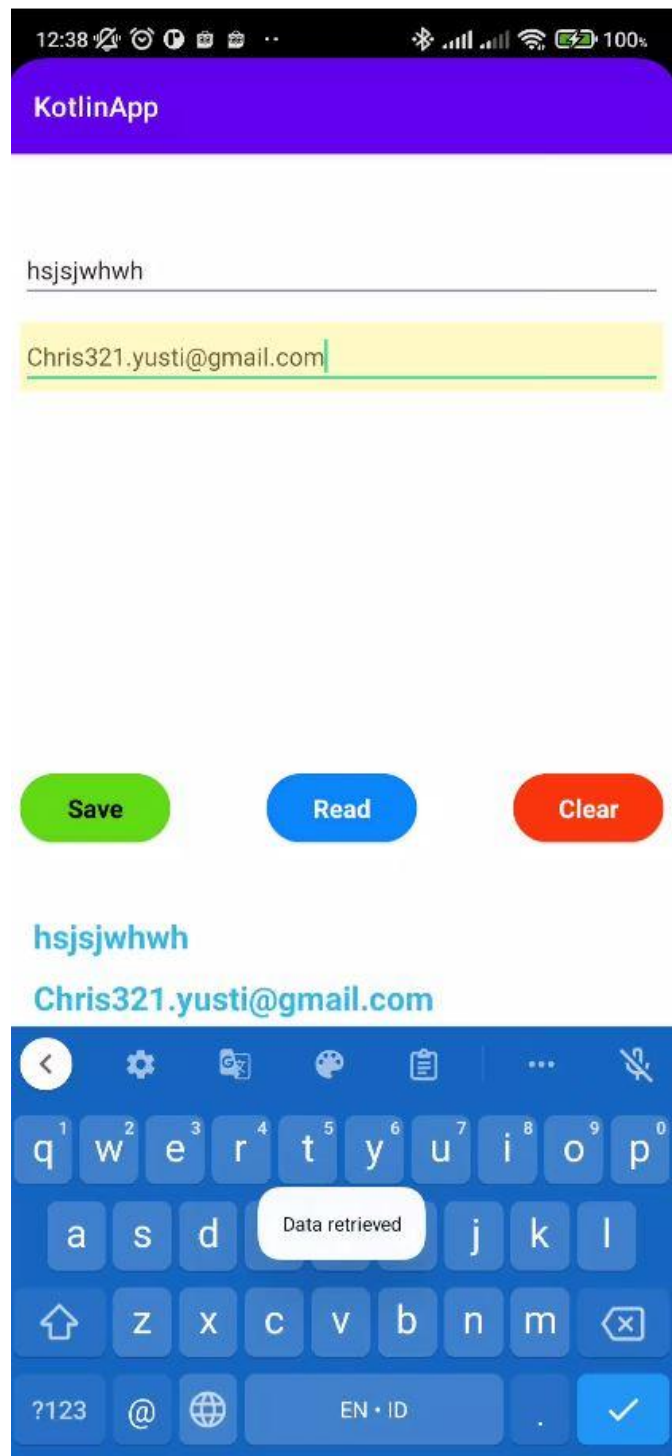
d. Silakan jalankan hasilnya seperti berikut :



Save Data :



Read Data :



Delete Data :

