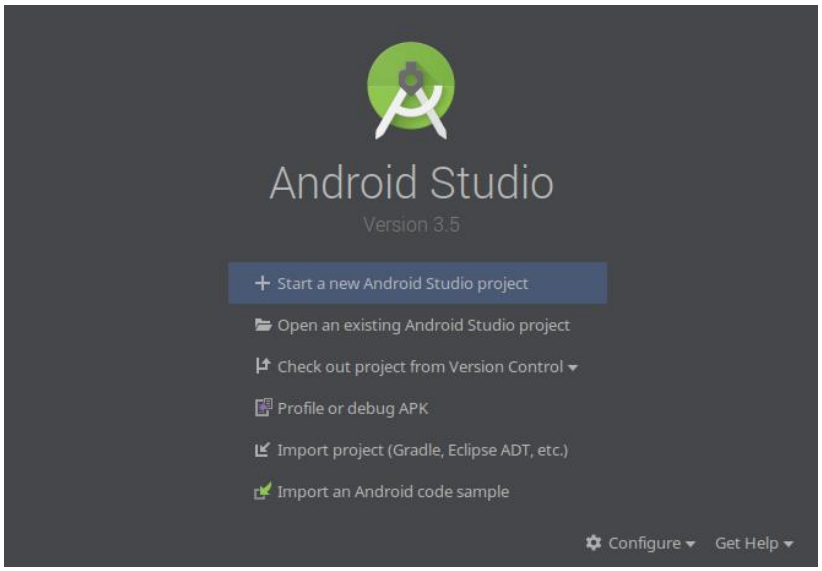
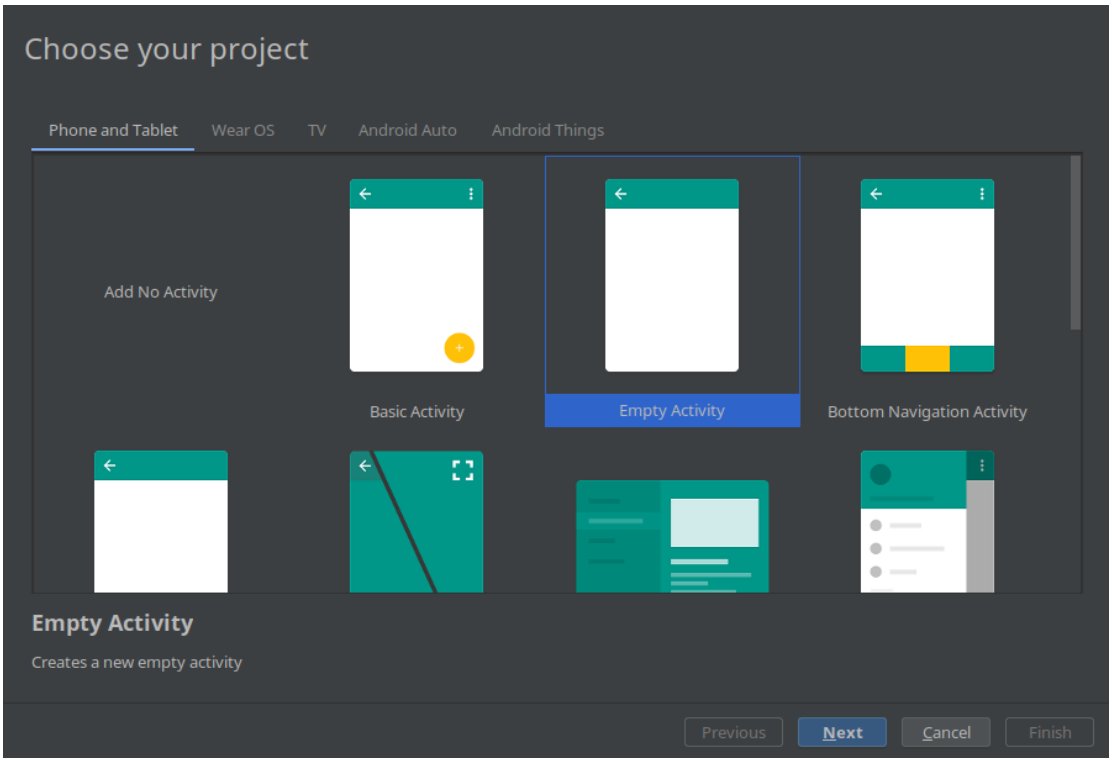
# Praktikum 7

# Parameter Passing

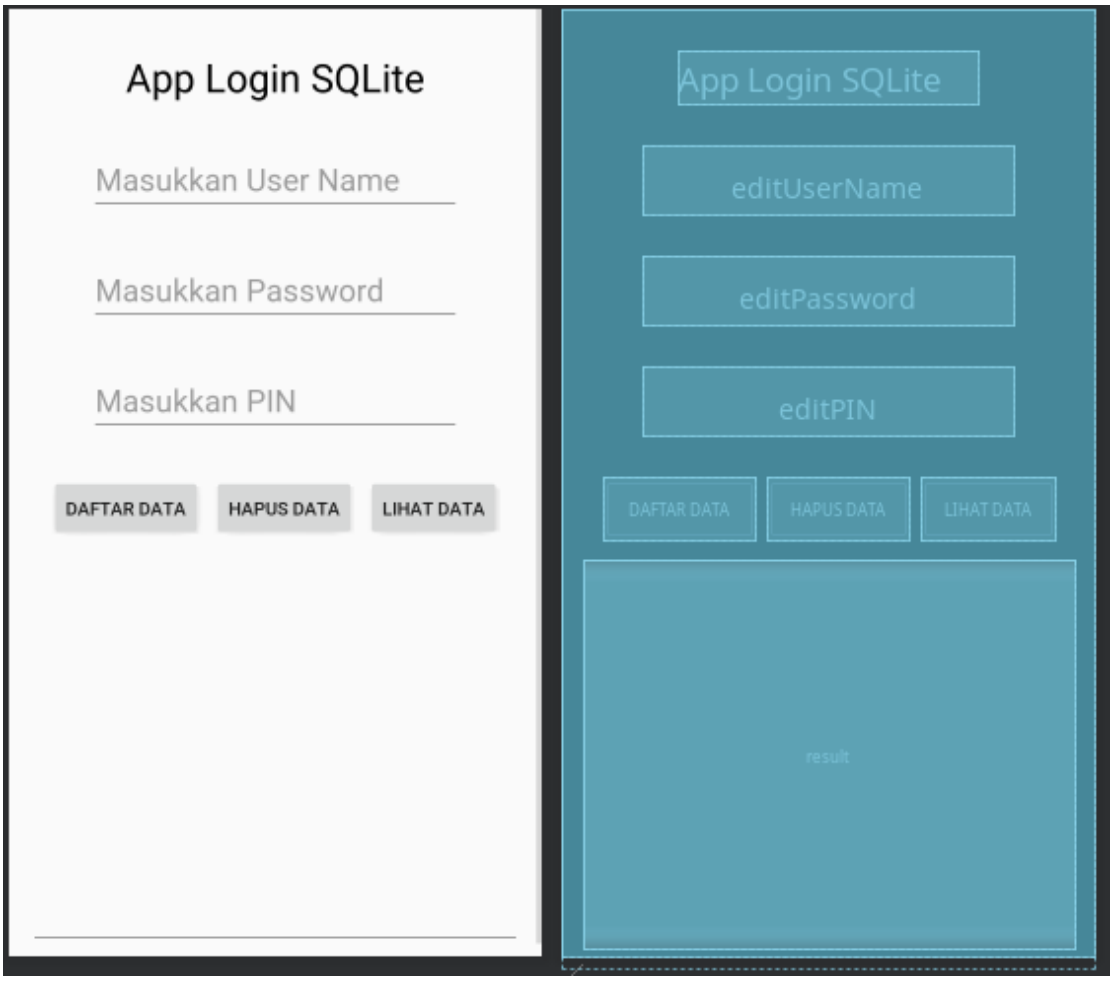
1. Buatlah proyek baru dengan Android Studio



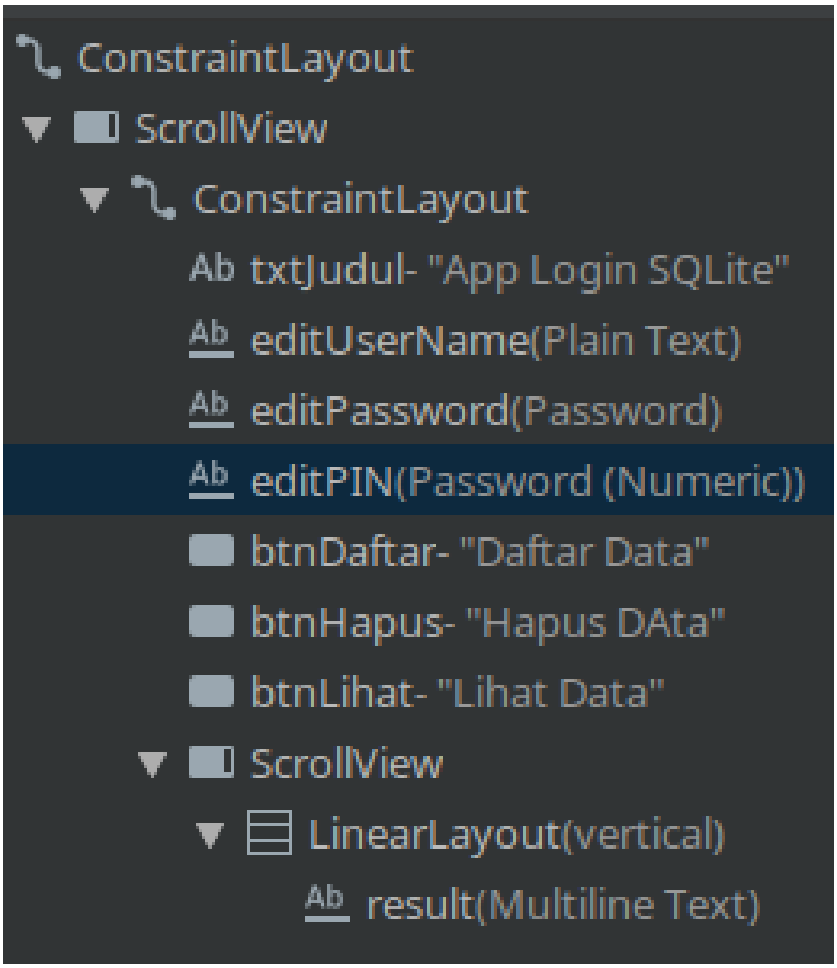

2. Pilih Empty Activity untuk memulai aplikasi baru



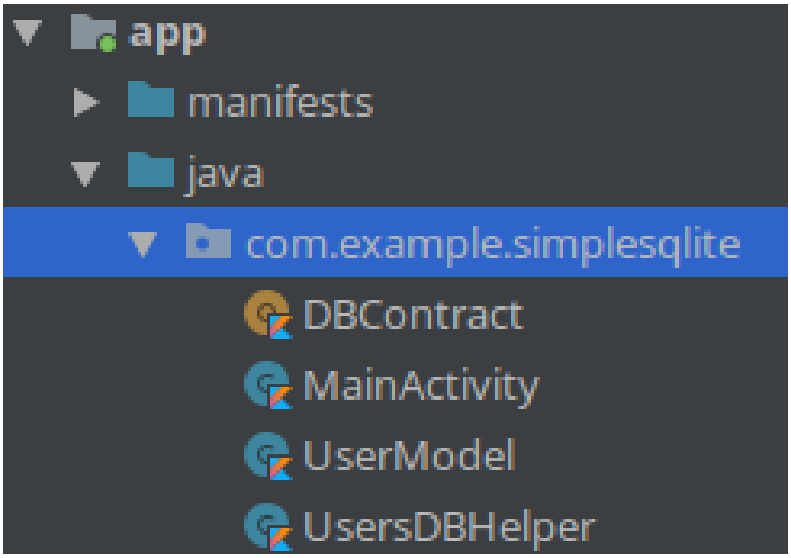
3. Isikan nama aplikasi dan lokasinya
4. Buatlah tampilan aplikasi seperti berikut, setelah fungsi background selesai:




5. Nama variabel yang dianjurkan agar mudah diingat




6. Buatlah File-File Berikut



7. Lalu lanjutkan dengan koding fungsional

**UserModel.kt**

```
package com.example.simplesqlite

class UserModel (
    val username: String,
    val password: String,
    val pin: String
)
```

**DBContract.kt**

```
package com.example.simplesqlite

import android.provider.BaseColumns

object DBContract {

    // Definisi Konten Tabel
    class UserEntry : BaseColumns {
        companion object {
            val TABLE_NAME = "users"
            val COLUMN_USER_NAME = "username"
            val COLUMN_PASSWORD = "password"
            val COLUMN_PIN = "pin"
        }
    }
}
```

**UsersDBHelper.kt**

```
package com.example.simplesqlite

import android.content.ContentValues
import android.content.Context
import android.database.Cursor
import android.database.sqlite.SQLiteConstraintException
import android.database.sqlite.SQLiteDatabase
import android.database.sqlite.SQLiteException
import android.database.sqlite.SQLiteOpenHelper

import java.util.ArrayList

class UsersDBHelper(context: Context) : SQLiteOpenHelper(context, DATA-
BASE_NAME, null, DATABASE_VERSION) {

    override fun onCreate(db: SQLiteDatabase) {
        db.execSQL(SQL_CREATE_ENTRIES)
    }
```

```
override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {
    // This database is only a cache for online data, so its upgrade policy is
    // to simply to discard the data and start over
    db.execSQL(SQL_DELETE_ENTRIES)
    onCreate(db)
}

override fun onDowngrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {
    onUpgrade(db, oldVersion, newVersion)
}

// Metode Tambah User
@Throws(SQLiteConstraintException::class)
fun insertUser(user: UserModel): Boolean {
    // Gets the data repository in write mode
    val db = writableDatabase

    // Mapping Data ke Tabel
    val values = ContentValues()
    values.put(DBContract.UserEntry.COLUMN_USER_NAME, user.username)
    values.put(DBContract.UserEntry.COLUMN_PASSWORD, user.password)
    values.put(DBContract.UserEntry.COLUMN_PIN, user.pin)

    // Insert the new row, returning the primary key value of the new row
    val newRowId = db.insert(DBContract.UserEntry.TABLE_NAME, null, values)

    return true
}

// Metode Hapus User
@Throws(SQLiteConstraintException::class)
fun deleteUser(userid: String): Boolean {
    val db = writableDatabase
    val selection = DBContract.UserEntry.COLUMN_USER_NAME + " LIKE ?"
    val selectionArgs = arrayOf(userid)
    db.delete(DBContract.UserEntry.TABLE_NAME, selection, selectionArgs)

    return true
}

// Membaca Semua Data
fun readAllUsers(): ArrayList<UserModel> {
    val users = ArrayList<UserModel>()
    val db = writableDatabase
    var cursor: Cursor? = null
```

```
        try {
            cursor = db.rawQuery("select * from " + DBContract.UserEntry.TABLE_NAME,
null)
        } catch (e: SQLiteException) {
            db.execSQL(SQL_CREATE_ENTRIES)
            return ArrayList()
        }

        var userid: String
        var name: String
        var age: String
        if (cursor!!.moveToFirst()) {
            while (cursor.isAfterLast == false) {
                userid = cursor.getString(cursor.getColumnIndex(DBContract.UserEntry.COL-
UMN_USER_NAME))
                name = cursor.getString(cursor.getColumnIndex(DBContract.UserEntry.COL-
UMN_PASSWORD))
                age = cursor.getString(cursor.getColumnIndex(DBContract.UserEntry.COL-
UMN_PIN))

                users.add(UserModel(userid, name, age))
                cursor.moveToNext()
            }
        }
        return users
    }

    // Entry SQL
    companion object {
        val DATABASE_VERSION = 1
        val DATABASE_NAME = "FeedReader.db"

        private val SQL_CREATE_ENTRIES =
            "CREATE TABLE " + DBContract.UserEntry.TABLE_NAME + " (" +
                    DBContract.UserEntry.COLUMN_USER_NAME + " TEXT PRIMARY
KEY," +
                    DBContract.UserEntry.COLUMN_PASSWORD + " TEXT," +
                    DBContract.UserEntry.COLUMN_PIN + " TEXT)"

        private val SQL_DELETE_ENTRIES = "DROP TABLE IF EXISTS " + DBCon-
tract.UserEntry.TABLE_NAME
    }

}
```

8. Ubahlah Kode dari MainActivity.kt

**MainActivity.kt**

```
package com.example.simplesqlite

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.EditText
import android.widget.TextView
// import kotlinx.android.synthetic.main.activity_main.*

class MainActivity : AppCompatActivity() {

    lateinit var usersDBHelper : UsersDBHelper

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

        val btnDaftar = findViewById<Button>(R.id.btnDaftar)
        val btnHapus = findViewById<Button>(R.id.btnHapus)
        val btnLihat = findViewById<Button>(R.id.btnLihat)

        // Inisialisasi DB
        usersDBHelper = UsersDBHelper(this)

        var resultText = findViewById<EditText>(R.id.result)

        // Aksi Daftar User
        btnDaftar.setOnClickListener {
            var editUserName = findViewById<EditText>(R.id.editUserName)
            var editPassword = findViewById<EditText>(R.id.editPassword)
            var editPin = findViewById<EditText>(R.id.editPIN)

            var username = editUserName.text.toString()
            var password = editPassword.text.toString()
            var pin = editPin.text.toString()

            var result = usersDBHelper.insertUser(UserModel(username = username,pass-
word = password,pin = pin))

            // Bersihkan Entry
            resultText.setText("Added user : "+ username)

            editUserName.setText("")
            editPassword.setText("")
            editPin.setText("")
```

```
        }

        btnHapus.setOnClickListener {
            var editUserName = findViewById<EditText>(R.id.editUserName)

            var username = editUserName.text.toString()

            val result = usersDBHelper.deleteUser(username)
            resultText.setText("Deleted user : "+result)
        }

        btnLihat.setOnClickListener {
            var users = usersDBHelper.readAllUsers()
            var out : String = ""
            users.forEach {
                out += it.username.toString() + "\n" + it.password.toString() + "\n" +
it.pin.toString() + "\n"
            }
            resultText.setText("Result:\n" + out)
        }

    }
}
```

9. Kompile Kode ke Emulator