



# Jetpack Compose

---

PRAKTIKUM PEMROGRAMAN MOBILE #4

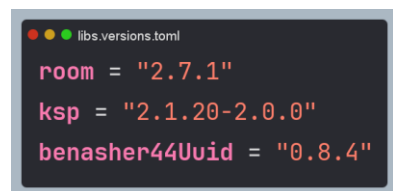
Wanda Gusdya Purnama  
TEKNIK INFORMATIKA UNIVERSITAS PASUNDAN  
2025

# Pendahuluan

Android Room adalah salah satu komponen dari Android Architecture Components yang dirancang untuk membuat interaksi dengan database SQLite menjadi lebih mudah dan aman. Dengan menggunakan Room, pengembang dapat mengakses database melalui objek Jawa/Kotlin yang merepresentasikan tabel dan entitas, meminimalisir kesalahan penulisan query SQL secara manual, serta mendukung fitur seperti migrasi database saat skema berubah. Room juga menyediakan kemampuan untuk melakukan query secara reaktif menggunakan LiveData dan Flow, yang memungkinkan aplikasi untuk memperbarui UI secara otomatis dengan data yang baru.

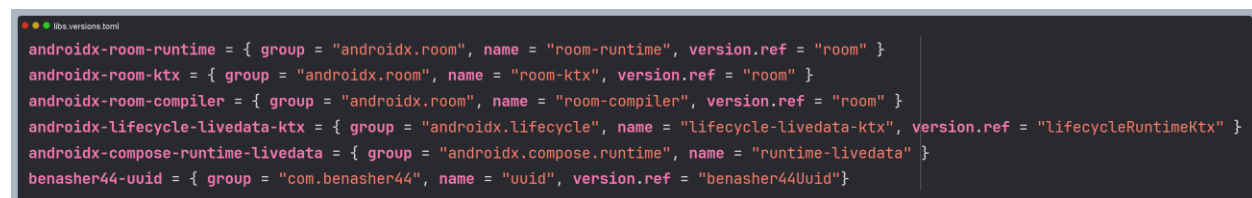
## Latihan 1

Pada latihan ini, kita akan mempelajari kemampuan library Android Room dengan membuat aplikasi untuk menyimpan catatan. Buatlah project dengan nama MyNote sesuai dengan langkah-langkah pembuatan project di Android Studio yang telah dipelajari. Pada bagian [versions] di file libs.versions.toml, tambahkan kode berikut



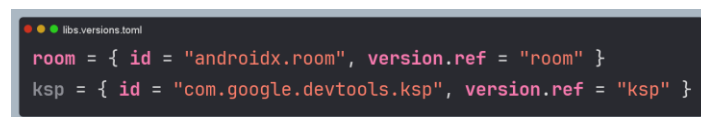
```
room = "2.7.1"
ksp = "2.1.20-2.0.0"
benasher44Uuid = "0.8.4"
```

Kemudian, di bagian [libraries], tambahkan kode berikut



```
androidx-room-runtime = { group = "androidx.room", name = "room-runtime", version.ref = "room" }
androidx-room-ktx = { group = "androidx.room", name = "room-ktx", version.ref = "room" }
androidx-room-compiler = { group = "androidx.room", name = "room-compiler", version.ref = "room" }
androidx-lifecycle-livedata-ktx = { group = "androidx.lifecycle", name = "lifecycle-livedata-ktx", version.ref = "lifecycleRuntimeKtx" }
androidx-compose-runtime-livedata = { group = "androidx.compose.runtime", name = "runtime-livedata" }
benasher44-uuid = { group = "com.benasher44", name = "uuid", version.ref = "benasher44Uuid" }
```

Kemudian, di bagian [plugins], tambahkan kode berikut



```
room = { id = "androidx.room", version.ref = "room" }
ksp = { id = "com.google.devtools.ksp", version.ref = "ksp" }
```

Sekarang, file libs.versions.toml akan terlihat seperti berikut

```

libx.versions.toml

[versions]
agp = "8.9.2"
kotlin = "2.1.20"
coreKtx = "1.16.0"
junit = "4.13.2"
junitVersion = "1.2.1"
espressoCore = "3.6.1"
lifecycleRuntimeKtx = "2.8.7"
activityCompose = "1.10.1"
composeBom = "2025.04.01"
room = "2.7.1"
ksp = "2.1.20-2.0.0"
benasher44Uuid = "0.8.4"

[libraries]
androidx-core-ktx = { group = "androidx.core", name = "core-ktx", version.ref = "coreKtx" }
junit = { group = "junit", name = "junit", version.ref = "junit" }
androidx-junit = { group = "androidx.test.ext", name = "junit", version.ref = "junitVersion" }
androidx-espresso-core = { group = "androidx.test.espresso", name = "espresso-core", version.ref = "espressoCore" }
androidx-lifecycle-runtime-ktx = { group = "androidx.lifecycle", name = "lifecycle-runtime-ktx", version.ref = "lifecycleRuntimeKtx" }
androidx-activity-compose = { group = "androidx.activity", name = "activity-compose", version.ref = "activityCompose" }
androidx-compose-bom = { group = "androidx.compose", name = "compose-bom", version.ref = "composeBom" }
androidx-ui = { group = "androidx.compose.ui", name = "ui" }
androidx-ui-graphics = { group = "androidx.compose.ui", name = "ui-graphics" }
androidx-ui-tooling = { group = "androidx.compose.ui", name = "ui-tooling" }
androidx-ui-tooling-preview = { group = "androidx.compose.ui", name = "ui-tooling-preview" }
androidx-ui-test-manifest = { group = "androidx.compose.ui", name = "ui-test-manifest" }
androidx-ui-test-junit4 = { group = "androidx.compose.ui", name = "ui-test-junit4" }
androidx-material3 = { group = "androidx.compose.material3", name = "material3" }
androidx-room-runtime = { group = "androidx.room", name = "room-runtime", version.ref = "room" }
androidx-room-ktx = { group = "androidx.room", name = "room-ktx", version.ref = "room" }
androidx-room-compiler = { group = "androidx.room", name = "room-compiler", version.ref = "room" }
androidx-lifecycle-livedata-ktx = { group = "androidx.lifecycle", name = "lifecycle-livedata-ktx", version.ref = "lifecycleRuntimeKtx" }
androidx-compose-runtime-livedata = { group = "androidx.compose.runtime", name = "runtime-livedata" }
benasher44-uuid = { group = "com.benasher44", name = "uuid", version.ref = "benasher44Uuid" }

[plugins]
android-application = { id = "com.android.application", version.ref = "agp" }
kotlin-android = { id = "org.jetbrains.kotlin.android", version.ref = "kotlin" }
kotlin-compose = { id = "org.jetbrains.kotlin.plugin.compose", version.ref = "kotlin" }
room = { id = "androidx.room", version.ref = "room" }
ksp = { id = "com.google.devtools.ksp", version.ref = "ksp" }

```

Lakukan sinkronisasi project gradle, kemudian buka file build.gradle.kts (Project: MyNote) dan tambahkan kode berikut di bagian plugins

```

build.gradle.kts

alias(libs.plugins.ksp) apply false
alias(libs.plugins.room) apply false

```

Sehingga file build.gradle.kts (Project: MyNote) akan terlihat seperti berikut

```

build.gradle.kts
// Top-level build file where you can add configuration options common to all sub-projects/modules.
plugins {
    alias(libs.plugins.android.application) apply false
    alias(libs.plugins.kotlin.android) apply false
    alias(libs.plugins.kotlin.compose) apply false
    alias(libs.plugins.ksp) apply false
    alias(libs.plugins.room) apply false
}

```

Selanjutnya, buka file build.gradle.kts (Module: app) dan tambahkan kode berikut pada bagian plugins

```

build.gradle.kts
alias(libs.plugins.ksp)
alias(libs.plugins.room)

```

Pada bagian android, tambahkan kode berikut

```

build.gradle.kts
room {
    schemaDirectory("$projectDir/schemas")
}

```

Kemudian di bagian dependencies, tambahkan kode berikut

```

build.gradle.kts
implementation(libs.androidx.lifecycle.livedata.ktx)
implementation(libs.androidx.compose.runtime.livedata)
implementation(libs.androidx.room.runtime)
implementation(libs.androidx.room.ktx)
ksp(libs.androidx.room.compiler)
implementation(libs.benasher44.uuid)

```

Sehingga keseluruhan file build.gradle akan terlihat seperti berikut

● ● ● build.gradle.kts

```
plugins {  
    alias(libs.plugins.android.application)  
    alias(libs.plugins.kotlin.android)  
    alias(libs.plugins.kotlin.compose)  
    alias(libs.plugins.ksp)  
    alias(libs.plugins.room)  
}  
  
android {  
    namespace = "id.ac.unpas.mynote"  
    compileSdk = 36  
  
    defaultConfig {  
        applicationId = "id.ac.unpas.mynote"  
        minSdk = 24  
        targetSdk = 36  
        versionCode = 1  
        versionName = "1.0"  
  
        testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"  
    }  
  
    buildTypes {  
        release {  
            isMinifyEnabled = false  
            proguardFiles(  
                getDefaultProguardFile("proguard-android-optimize.txt"),  
                "proguard-rules.pro"  
            )  
        }  
    }  
  
    compileOptions {  
        sourceCompatibility = JavaVersion.VERSION_11  
        targetCompatibility = JavaVersion.VERSION_11  
    }  
}
```

```

build.gradle.kts

kotlinOptions {
    jvmTarget = "11"
}

buildFeatures {
    compose = true
}

room {
    schemaDirectory("$projectDir/schemas")
}

}

dependencies {

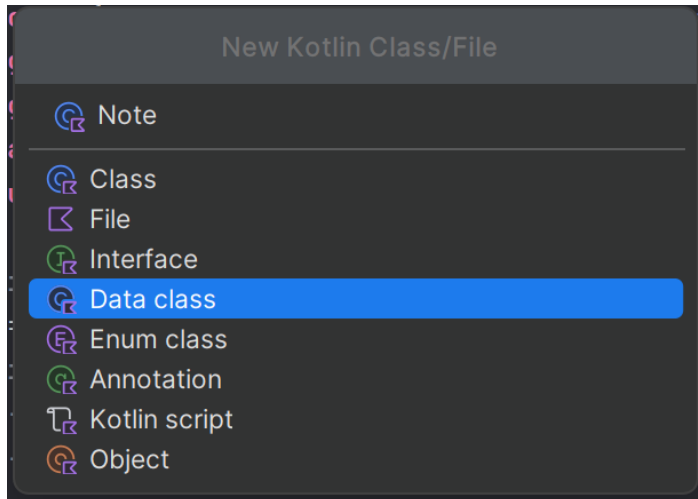
    implementation(libs.androidx.core.ktx)
    implementation(libs.androidx.lifecycle.runtime.ktx)
    implementation(libs.androidx.activity.compose)
    implementation(platform(libs.androidx.compose.bom))
    implementation(libs.androidx.ui)
    implementation(libs.androidx.ui.graphics)
    implementation(libs.androidx.ui.tooling.preview)
    implementation(libs.androidx.material3)
    testImplementation(libs.junit)
    androidTestImplementation(libs.androidx.junit)
    androidTestImplementation(libs.androidx.espresso.core)
    androidTestImplementation(platform(libs.androidx.compose.bom))
    androidTestImplementation(libs.androidx.ui.test.junit4)
    debugImplementation(libs.androidx.ui.tooling)
    debugImplementation(libs.androidx.ui.test.manifest)

    implementation(libs.androidx.lifecycle.livedata.ktx)
    implementation(libs.androidx.compose.runtime.livedata)
    implementation(libs.androidx.room.runtime)
    implementation(libs.androidx.room.ktx)
    ksp(libs.androidx.room.compiler)
    implementation(libs.benasher44.uuid)
}

```

Lakukan sinkronisasi project gradle sekali lagi.

Setelah proses sinkronisasi selesai, buat paket baru dengan nama models, kemudian buat kelas kotlin baru dengan nama Note dan jenis Data class



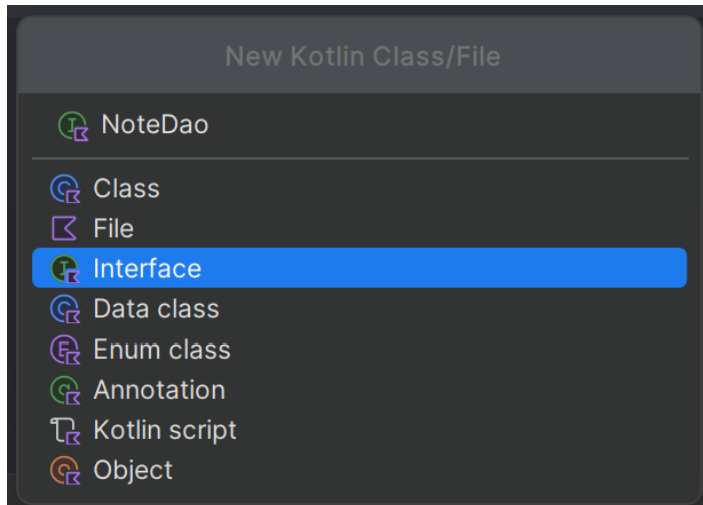
Kelas Note adalah kelas Entity yang merepresentasikan sebuah table bernama notes. Primary key dari table ini bertipe data String dengan menggunakan uuid. Tambahkan kode berikut pada kelas Note yang baru saja dibuat.

```
Note.kt
package id.ac.unpas.mynote.models

import androidx.room.Entity
import androidx.room.PrimaryKey

@Entity(tableName = "notes")
data class Note(
    @PrimaryKey
    val id: String,
    val title: String,
    val description: String
)
```

Selanjutnya, buat paket dao kemudian buat interface kotlin dengan nama NoteDao



DAO adalah singkatan dari Data Access Object, yaitu object yang berguna untuk mengakses data (biasanya melalui query). Pengembang hanya perlu membuat interface dengan anotasi yang telah disediakan oleh Android Room, sehingga dapat mengurangi waktu dan kesalahan saat membuat query. Ketikkan kode berikut untuk interface NoteDao.

```
package id.ac.unpas.mynote.dao

import androidx.lifecycle.LiveData
import androidx.room.Dao
import androidx.room.Delete
import androidx.room.Insert
import androidx.room.Query
import id.ac.unpas.mynote.models.Note

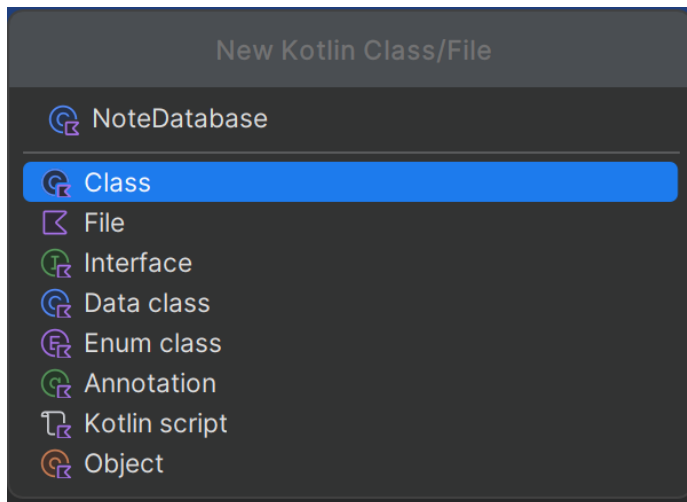
@Dao
interface NoteDao {
    @Query("SELECT * FROM notes")
    fun getAllNotes(): LiveData<List<Note>>

    @Insert
    suspend fun insertNote(note: Note)

    @Delete
    suspend fun deleteNote(note: Note)
}
```



Kemudian, di package utama, buat class dengan nama NoteDatabase. Kelas ini akan digunakan untuk mendefinisikan database. Semua kelas Entity dan DAO harus didaftarkan di sini.



Tulis kode berikut pada kelas NoteDatabase

```

NoteDatabase.kt

package id.ac.unpas.mynote

import android.content.Context
import androidx.room.Database
import androidx.room.Room
import androidx.room.RoomDatabase
import id.ac.unpas.mynote.dao.NoteDao
import id.ac.unpas.mynote.models.Note

@Database(entities = [Note::class], version = 1)
abstract class NoteDatabase : RoomDatabase() {
    abstract fun noteDao(): NoteDao

    companion object {
        @Volatile
        private var INSTANCE: NoteDatabase? = null

        fun getDatabase(context: Context): NoteDatabase {
            return INSTANCE ?: synchronized(this) {
                Room.databaseBuilder(
                    context.applicationContext,
                    NoteDatabase::class.java,
                    "note_database"
                ).build().also { INSTANCE = it }
            }
        }
    }
}

```

Selanjutnya, buat Composable bernama NoteScreen sebagai UI untuk menginputkan dan melihat data Notes. Composable ini mengakses data menggunakan LiveData, sehingga pengguna tidak perlu melakukan request ulang setelah melakukan update data. Perlu diperhatikan, setiap fungsi suspend dapat memblokir thread UI sehingga harus dijalankan di dalam scope terpisah. Ketikkan kode berikut pada file NoteScreen.

● ● ● NoteScreen.kt

```
package id.ac.unpas.mynote

import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Spacer
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.height
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.lazy.LazyColumn
import androidx.compose.foundation.lazy.items
import androidx.compose.material3.Button
import androidx.compose.material3.Card
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Text
import androidx.compose.material3.TextField
import androidx.compose.runtime.Composable
import androidx.compose.runtime.getValue
import androidx.compose.runtime.livedata.observeAsState
import androidx.compose.runtime.mutableStateOf
import androidx.compose.runtime.remember
import androidx.compose.runtime.rememberCoroutineScope
import androidx.compose.runtime.setValue
import androidx.compose.ui.Modifier
import androidx.compose.ui.platform.LocalContext
import androidx.compose.ui.unit.dp
import androidx.lifecycle.LiveData
import com.benasher44.uuid.uuid4
import id.ac.unpas.mynote.models.Note
import kotlinx.coroutines.launch
```

```

NoteScreen.kt

@Composable
fun NoteScreen(modifier: Modifier) {
    val context = LocalContext.current
    val dao = NoteDatabase.getDatabase(context).noteDao()
    val list : LiveData<List<Note>> = dao.getAllNotes()
    val notes: List<Note> by list.observeAsState(initial = listOf())

    var title by remember { mutableStateOf("") }
    var description by remember { mutableStateOf("") }
    val scope = rememberCoroutineScope()

    Column(modifier = modifier.padding(16.dp)) {
        TextField(
            value = title,
            onValueChange = { title = it },
            label = { Text("Title") },
            modifier = Modifier.fillMaxWidth()
        )
        Spacer(Modifier.height(8.dp))
        TextField(
            value = description,
            onValueChange = { description = it },
            label = { Text("Description") },
            modifier = Modifier.fillMaxWidth()
        )
        Spacer(Modifier.height(8.dp))
        Button(
            onClick = {
                if (title.isNotEmpty() && description.isNotEmpty()) {
                    scope.launch {
                        dao.insertNote(Note(uuid4().toString(), title, description))
                        title = ""
                        description = ""
                    }
                }
            },
            modifier = Modifier.fillMaxWidth()
        ) {
            Text("Save Note")
        }

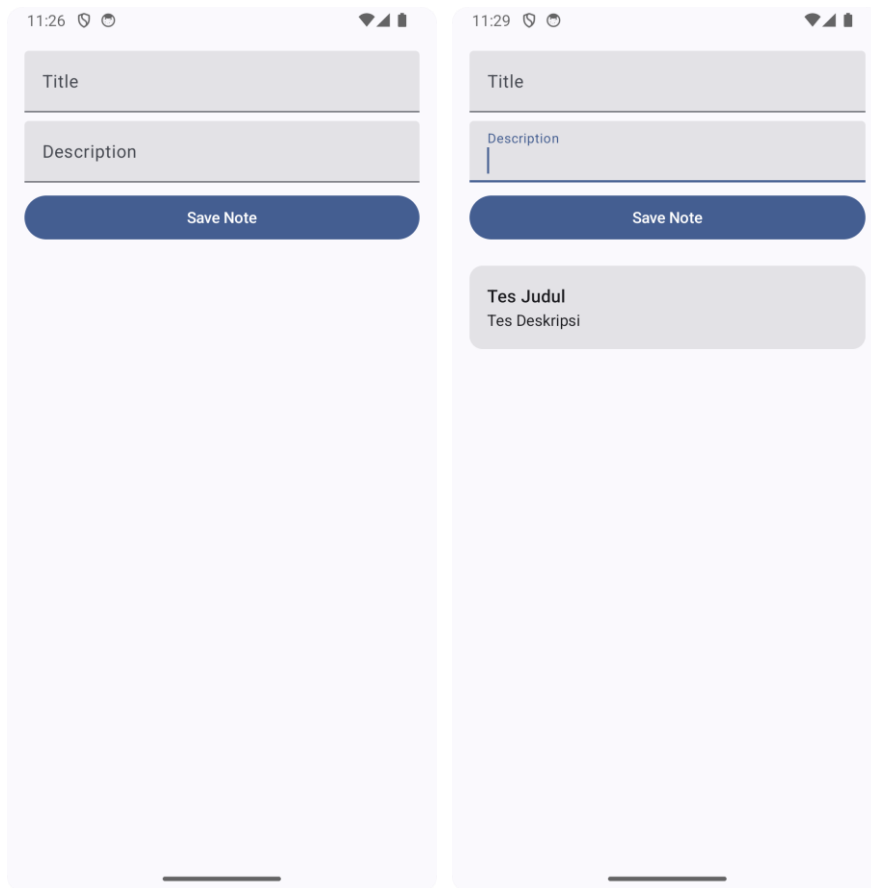
        Spacer(Modifier.height(16.dp))
    }
}

```

```
NoteScreen.kt

LazyColumn {
    items(notes) { note ->
        Card(
            modifier = Modifier
                .fillMaxWidth()
                .padding(vertical = 4.dp),
            onClick = {
                scope.launch {
                    dao.deleteNote(note)
                }
            }
        ) {
            Column(modifier = Modifier.padding(16.dp)) {
                Text(note.title, style = MaterialTheme.typography.titleMedium)
                Text(note.description, style = MaterialTheme.typography.bodyMedium)
            }
        }
    }
}
}
```

Jalankan aplikasi dan amati hasilnya.



## Latihan 2

Tambahkan fitur update untuk aplikasi MyNote. Anda dapat menambahkan 2 tombol untuk trigger edit dan delete atau semacamnya.