**Practical-11**

**AIM:** Create Note Android Application that can add Note, edit Note, delete Note, and set reminder date & time of note. By using Broadcast Receiver, AlarmManager set reminder of note. By using SQLite, store all notes data.

**Code:**

**Activity\_main.xml:**

<?xml version="1.0" encoding="utf-8"?>  
<androidx.coordinatorlayout.widget.CoordinatorLayout 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="com.example.mad\_20012021033\_p\_11.MainActivity">  
  
 <com.google.android.material.appbar.AppBarLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:fitsSystemWindows="true">  
  
 <com.google.android.material.appbar.MaterialToolbar  
 android:id="@+id/toolbar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="?attr/actionBarSize"  
 app:layout\_scrollFlags="scroll|enterAlways|snap"  
 app:title="mad\_20012021033\_p\_11" />  
 </com.google.android.material.appbar.AppBarLayout>  
  
 <androidx.constraintlayout.widget.ConstraintLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_marginBottom="?attr/actionBarSize"  
 app:layout\_behavior="@string/appbar\_scrolling\_view\_behavior">  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/listView1"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 app:layoutManager="androidx.recyclerview.widget.LinearLayoutManager"  
 tools:itemCount="5"  
 tools:listitem="@layout/note\_item\_view" />  
  
 </androidx.constraintlayout.widget.ConstraintLayout>  
 <com.google.android.material.floatingactionbutton.FloatingActionButton  
 android:id="@+id/fab"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="bottom|end"  
 android:layout\_marginEnd="16dp"  
 android:layout\_marginBottom="56dp"  
 app:srcCompat="@android:drawable/ic\_input\_add" />  
</androidx.coordinatorlayout.widget.CoordinatorLayout>

**Activity\_note\_view.xml:**

<?xml version="1.0" encoding="utf-8"?>  
<androidx.coordinatorlayout.widget.CoordinatorLayout 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="com.example.madpractical11\_20012011035.NoteViewActivity">  
  
 <com.google.android.material.appbar.AppBarLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:fitsSystemWindows="true">  
  
 <com.google.android.material.appbar.MaterialToolbar  
 android:id="@+id/toolbar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="?attr/actionBarSize"  
 app:layout\_scrollFlags="scroll|enterAlways|snap" />  
 </com.google.android.material.appbar.AppBarLayout>  
  
 <androidx.constraintlayout.widget.ConstraintLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_marginBottom="?attr/actionBarSize"  
 app:layout\_behavior="@string/appbar\_scrolling\_view\_behavior">  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:padding="16dp">  
  
 <TextView  
 android:id="@+id/note\_title"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ellipsize="end"  
 android:gravity="center\_vertical"  
 android:textAppearance="@style/TextAppearance.AppCompat"  
 android:textSize="20sp"  
 android:textStyle="bold"  
 tools:text="This is Note Title but some time it is bigger than think." />  
  
 <TextView  
 android:id="@+id/note\_subtitle"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ellipsize="end"  
 android:gravity="center\_vertical"  
 android:textSize="16sp"  
 tools:text="SubTitle (optional)" />  
  
 <TextView  
 android:id="@+id/note\_content"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="8dp"  
 android:layout\_marginBottom="8dp"  
 android:autoLink="all"  
 android:ellipsize="end"  
 android:textAppearance="@style/TextAppearance.AppCompat"  
 android:textSize="18sp"  
 tools:text="Description" />  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal">  
 <TextView  
 android:layout\_gravity="center\_vertical"  
 android:id="@+id/note\_date"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 tools:text="Date and time" />  
  
 </LinearLayout>  
  
 <TextView  
 android:layout\_gravity="center\_vertical"  
 android:id="@+id/note\_reminder\_date\_time"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 tools:text="Reminder Date and time" />  
  
 </LinearLayout>  
 </androidx.constraintlayout.widget.ConstraintLayout>  
  
</androidx.coordinatorlayout.widget.CoordinatorLayout>

**Note\_edit\_view.xml:**

<?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"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 >  
 <androidx.core.widget.NestedScrollView  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 android:layout\_marginHorizontal="20dp"  
 android:layout\_marginVertical="20dp">  
 <androidx.constraintlayout.widget.ConstraintLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
 <com.google.android.material.textfield.TextInputLayout  
 android:id="@+id/textInputLayout2"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent">  
  
 <com.google.android.material.textfield.TextInputEditText  
 android:id="@+id/note\_title"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:drawableTint="?attr/colorPrimary"  
 android:drawableEnd="@drawable/ic\_baseline\_event\_note\_24"  
 android:hint="Note Title" />  
 </com.google.android.material.textfield.TextInputLayout>  
  
 <com.google.android.material.textfield.TextInputLayout  
 android:id="@+id/textInputLayout4"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textInputLayout2">  
  
 <com.google.android.material.textfield.TextInputEditText  
 android:id="@+id/note\_subTitle"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:drawableTint="?attr/colorPrimary"  
 android:drawableEnd="@drawable/ic\_baseline\_event\_note\_24"  
 android:hint="Note Sub Title" />  
 </com.google.android.material.textfield.TextInputLayout>  
  
 <com.google.android.material.textfield.TextInputLayout  
 android:id="@+id/textInputLayout3"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textInputLayout4">  
  
 <com.google.android.material.textfield.TextInputEditText  
 android:id="@+id/note\_description"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:drawableEnd="@drawable/ic\_baseline\_event\_note\_24"  
 android:gravity="top"  
 android:drawableTint="?attr/colorPrimary"  
 android:hint="Note Description"  
 android:minLines="5" />  
 </com.google.android.material.textfield.TextInputLayout>  
 <LinearLayout  
 android:orientation="horizontal"  
 android:id="@+id/layoutReminder"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textInputLayout3">  
 <TextView  
 android:visibility="gone"  
 android:text="Reminder:"  
 android:textSize="30sp"  
 android:textStyle="bold"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"/>  
  
 <com.google.android.material.switchmaterial.SwitchMaterial  
 android:id="@+id/reminderSwitch"  
 android:layout\_width="0dp"  
 android:gravity="end|center\_vertical"  
 android:textSize="20sp"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="Set Reminder" />  
  
 </LinearLayout>  
 <TimePicker  
 android:id="@+id/reminderTime"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/layoutReminder" />  
 </androidx.constraintlayout.widget.ConstraintLayout>  
 </androidx.core.widget.NestedScrollView>  
</androidx.constraintlayout.widget.ConstraintLayout>

**Note\_item\_view.xml:**

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools">  
 <com.google.android.material.card.MaterialCardView  
 android:id="@+id/card\_note"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginVertical="10dp"  
 android:layout\_marginHorizontal="20dp"  
 style="?attr/materialCardViewElevatedStyle"  
 app:cardElevation="10dp"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent"  
 android:layout\_marginBottom="30dp"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 >  
 <LinearLayout  
 android:orientation="horizontal"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content">  
 <LinearLayout  
 android:layout\_weight="0.9"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical"  
 android:padding="16dp">  
  
 <TextView  
 android:id="@+id/note\_title"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ellipsize="end"  
 android:gravity="center\_vertical"  
 android:textAppearance="@style/TextAppearance.AppCompat"  
 android:textSize="20sp"  
 android:textStyle="bold"  
 android:maxLines="1"  
 tools:text="This is Note Title but some time it is bigger than think." />  
  
 <TextView  
 android:id="@+id/note\_subtitle"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:ellipsize="end"  
 android:gravity="center\_vertical"  
 android:maxLines="1"  
 android:textSize="16sp"  
 tools:text="SubTitle" />  
  
 <TextView  
 android:visibility="gone"  
 android:id="@+id/note\_content"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="8dp"  
 android:layout\_marginBottom="8dp"  
 android:autoLink="all"  
 android:ellipsize="end"  
 android:maxLines="4"  
 android:textAppearance="@style/TextAppearance.AppCompat"  
 android:textSize="18sp"  
 tools:text="Description" />  
  
 <TextView  
 android:visibility="gone"  
 android:layout\_gravity="center\_vertical"  
 android:id="@+id/note\_date"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 tools:text="modified Date and time" />  
 <TextView  
 android:layout\_gravity="center\_vertical"  
 android:id="@+id/note\_reminder\_date\_time"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textColor="#FF3355"  
 android:textStyle="bold"  
 tools:text="Reminder Date and time" />  
 </LinearLayout>  
 <LinearLayout  
 android:layout\_gravity="center\_vertical"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal">  
 <ImageView  
 android:id="@+id/imgEdit"  
 android:padding="10dp"  
 android:layout\_marginEnd="10dp"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:src="@drawable/ic\_baseline\_edit\_24"  
 app:tint="@color/purple\_200" />  
 <ImageView  
 android:layout\_marginEnd="10dp"  
 android:id="@+id/imgDelete"  
 android:padding="10dp"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:src="@drawable/ic\_baseline\_delete\_24"  
 app:tint="#FF2244" />  
 </LinearLayout>  
 </LinearLayout>  
 </com.google.android.material.card.MaterialCardView>  
</androidx.constraintlayout.widget.ConstraintLayout>

**MainActivity.kt:**

package com.example.mad\_20012021033\_p\_11  
  
import android.os.Build  
import androidx.appcompat.app.AppCompatActivity  
import android.os.Bundle  
import android.util.Log  
import android.view.LayoutInflater  
import android.widget.Toast  
import androidx.appcompat.app.AlertDialog  
import androidx.core.view.WindowCompat  
import androidx.recyclerview.widget.DefaultItemAnimator  
import androidx.recyclerview.widget.LinearLayoutManager  
import androidx.recyclerview.widget.RecyclerView  
import com.example.mad\_20012021033\_p\_11.Note  
import com.example.mad\_20012021033\_p\_11.NoteMode  
import com.example.mad\_20012021033\_p\_11.databinding.ActivityMainBinding  
import com.example.mad\_20012021033\_p\_11.databinding.NoteEditViewBinding  
  
  
import java.util.ArrayList  
  
class MainActivity : AppCompatActivity() {  
 private lateinit var binding: ActivityMainBinding  
 private val TAG = "MainActivity"  
 private var listener: ((note: Note, baseListAdapter: NotesAdapter, mode: NoteMode, position: Int)->Unit)? =  
 **{** note: Note, \_: NotesAdapter, noteMode: NoteMode, pos: Int **->** note.modifiedTime = Note.getCurrentDateTime()  
 if (noteMode == NoteMode.*add*) {  
 if (!createNote(note))  
 Toast.makeText(this, "Enter Valid Note", Toast.*LENGTH\_SHORT*).show()  
 } else if (noteMode == NoteMode.*edit*) {  
 Log.i(TAG, "listener: Note:$note")  
 if (!updateNote(note, pos))  
 Toast.makeText(this, "Enter Valid Note", Toast.*LENGTH\_SHORT*).show()  
 }  
 **}** lateinit var db: DatabaseHelper  
 private val notesList: ArrayList<Note> = ArrayList<Note>()  
 lateinit var notesRecycleAdapter: NotesAdapter  
 override fun onCreate(savedInstanceState: Bundle?) {  
 WindowCompat.setDecorFitsSystemWindows(*window*, false)  
 super.onCreate(savedInstanceState)  
  
 binding = ActivityMainBinding.inflate(*layoutInflater*)  
 setContentView(binding.*root*)  
  
 db = DatabaseHelper(this)  
 notesList.addAll(db.allNotes)  
  
 notesRecycleAdapter = NotesAdapter(this, notesList)  
 val mLayoutManager: RecyclerView.LayoutManager = LinearLayoutManager(*applicationContext*)  
 binding.listView1.*layoutManager* = mLayoutManager  
 binding.listView1.*itemAnimator* = DefaultItemAnimator()  
 binding.listView1.setHasFixedSize(true)  
 binding.listView1.*adapter* = notesRecycleAdapter  
 setSupportActionBar(binding.toolbar)  
 binding.fab.setOnClickListener **{** showAlertDialog(  
 NoteMode.*add*, "Add Note",  
 Note("", "", "", Note.getCurrentDateTime()), -1, notesRecycleAdapter  
 )  
 **}** }  
 */\*\*  
 \* Inserting new note in db  
 \* and refreshing the list  
 \*/* private fun createNote(note: Note):Boolean {  
 if(!note.isValid())  
 return false  
 // inserting note in db and getting  
 // newly inserted note id  
 val id = db.insertNote(note)  
  
 // get the newly inserted note from db  
 val n: Note = db.getNote(id)  
 // adding new note to array list at 0 position  
 notesList.add(0, n)  
  
 // refreshing the list  
 notesRecycleAdapter.notifyItemInserted(0)  
 note.saveNote(this)  
 return true  
 }  
  
 */\*\*  
 \* Updating note in db and updating  
 \* item in the list by its position  
 \*/* private fun updateNote(note: Note, position: Int):Boolean {  
 if(!note.isValid())  
 return false  
 val n: Note = notesList[position]  
 n.changeValue(note)  
 Log.i(TAG, "updateNote: note:: $n")  
 // updating note in db  
 db.updateNote(n)  
  
 // refreshing the list  
 notesList[position] = n  
 notesRecycleAdapter.notifyItemChanged(position)  
 note.saveNote(this)  
 return true  
 }  
 */\*\*  
 \* Deleting note from SQLite and removing the  
 \* item from the list by its position  
 \*/* fun deleteNote(position: Int) {  
 // deleting the note from db  
 db.deleteNote(notesList[position])  
  
 // removing the note from the list  
 notesList.removeAt(position)  
 notesRecycleAdapter.notifyItemRemoved(position)  
 }  
 fun showAlertDialog(  
 mode: NoteMode,  
 dialogTitle: String,  
 note: Note,  
 position: Int,  
 baseListAdapter: NotesAdapter  
 ) {  
 val builder: AlertDialog.Builder = AlertDialog.Builder(this)  
 builder.setTitle(dialogTitle)  
 // set the custom layout  
 val binding = NoteEditViewBinding.inflate(LayoutInflater.from(this))  
 binding.noteTitle.setText(note.title)  
 binding.noteSubTitle.setText(note.subTitle)  
 binding.noteDescription.setText(note.Description)  
 binding.reminderSwitch.*isChecked* = note.isReminder  
 if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*M*) {  
 binding.reminderTime.*hour* = note.getHour()  
 binding.reminderTime.*minute* = note.getMinute()  
 }  
 else{  
 binding.reminderTime.*currentHour* = note.getHour()  
 binding.reminderTime.*currentMinute* = note.getMinute()  
 }  
  
 builder.setView(binding.*root*)  
 // add a button  
 builder.setPositiveButton(  
 "OK"  
 ) **{** \_, \_ **->** val newNote = Note(note)  
 newNote.title = binding.noteTitle.*text*.*toString*()  
 newNote.subTitle = binding.noteSubTitle.*text*.*toString*()  
 newNote.Description = binding.noteDescription.*text*.*toString*()  
 newNote.isReminder = binding.reminderSwitch.*isChecked* if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*M*)  
 newNote.remindertime = Note.getMillis(binding.reminderTime.*hour*, binding.reminderTime.*minute*)  
 else  
 newNote.remindertime = Note.getMillis(binding.reminderTime.*currentHour*, binding.reminderTime.*currentMinute*)  
 Log.i(TAG, "showAlertDialog: OK Button:: Note:$newNote")  
 listener?.invoke(newNote, baseListAdapter, mode, position)  
 **}** // create and show the alert dialog  
 val dialog: AlertDialog = builder.create()  
 dialog.show()  
 }  
}

**AlarmBroadcastreceiver.kt:**

package com.example.mad\_20012021033\_p\_11  
  
import android.app.Notification  
import android.app.NotificationChannel  
import android.app.NotificationManager  
import android.app.PendingIntent  
import android.content.BroadcastReceiver  
import android.content.Context  
import android.content.Intent  
import android.graphics.Color  
import android.os.Build  
import android.util.Log  
import androidx.core.app.NotificationCompat  
import com.example.mad\_20012021033\_p\_11.R  
import java.io.Serializable  
  
class AlarmBroadcastReceiver : BroadcastReceiver() {  
 val TAG = "AlarmBroadcastReceiver"  
 override fun onReceive(context: Context, intent: Intent) {  
 if(intent != null && context!=null) {  
 val note = Note()  
 note.id = intent.getIntExtra(Note.NOTE\_ID\_KEY, -1)  
 note.title = intent.getStringExtra(Note.NOTE\_TITLE\_KEY)!!  
 note.subTitle = intent.getStringExtra(Note.NOTE\_SUBTITLE\_KEY)!!  
 note.Description = intent.getStringExtra(Note.NOTE\_DESCRIPTION\_KEY)!!  
 note.modifiedTime = intent.getStringExtra(Note.NOTE\_MODIFIED\_TIME\_KEY)!!  
 note.remindertime = intent.getLongExtra(Note.NOTE\_REMINDER\_TIME\_KEY, 0)  
 note.isReminder = true  
 Log.i(TAG, "onReceive: Note: $note")  
 notificationDialog(  
 context,  
 NoteViewActivity::class.*java*,  
 note.title,  
 note.Description,  
 note  
 )  
 }  
 }  
 private fun notificationDialog(context: Context, cls: Class<\*>, title: String, descr: String,note: Note) {  
 val notificationManager = context.getSystemService(Context.*NOTIFICATION\_SERVICE*) as NotificationManager  
 val NOTIFICATION\_CHANNEL\_ID = "hitensadani"  
 if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*O*) {  
 val notificationChannel = NotificationChannel(  
 NOTIFICATION\_CHANNEL\_ID,  
 "Note Application", NotificationManager.*IMPORTANCE\_HIGH* )  
 // Configure the notification channel.  
 notificationChannel.*description* = "Note Application description"  
 notificationChannel.enableLights(true)  
 notificationChannel.*lightColor* = Color.*RED* notificationChannel.*vibrationPattern* = *longArrayOf*(0, 1000, 500, 1000)  
 notificationChannel.enableVibration(true)  
 notificationManager.createNotificationChannel(notificationChannel)  
 }  
 val notificationIntent = Intent(context, cls)  
 notificationIntent.addFlags(Intent.*FLAG\_ACTIVITY\_CLEAR\_TOP*)  
 //notification message will get at NotificationView  
 notificationIntent.putExtra("message", "This is a notification message")  
 notificationIntent.putExtra("Object", note as Serializable)  
 val pendingIntent = PendingIntent.getActivity(  
 context, Note.REMINDER\_REQUEST\_CODE, notificationIntent,  
 PendingIntent.*FLAG\_UPDATE\_CURRENT* )  
 val notificationBuilder = NotificationCompat.Builder(context, NOTIFICATION\_CHANNEL\_ID)  
 notificationBuilder.setAutoCancel(true)  
 .setDefaults(Notification.*DEFAULT\_ALL*)  
 .setWhen(System.currentTimeMillis())  
 .setSmallIcon(R.mipmap.*ic\_launcher*)  
 .setTicker("NoteApplication-HitenSadani") //.setPriority(Notification.PRIORITY\_MAX)  
 .setContentTitle(title)  
 .setContentText(descr)  
 .setContentInfo(descr + "Information")  
 .setAutoCancel(true)  
 .setContentIntent(pendingIntent)  
 notificationManager.notify(1, notificationBuilder.build())  
 }  
}

**DatabaseHelper.kt:**

package com.example.mad\_20012021033\_p\_11  
  
import android.content.ContentValues  
import android.content.Context  
import android.database.Cursor  
import android.database.sqlite.SQLiteDatabase  
import android.database.sqlite.SQLiteOpenHelper  
import java.util.ArrayList  
  
class DatabaseHelper(context: Context?) :  
 SQLiteOpenHelper(context, DATABASE\_NAME, null, DATABASE\_VERSION) {  
 companion object {  
 // Database Version  
 private const val DATABASE\_VERSION = 1  
 // Database Name  
 private const val DATABASE\_NAME = "notes\_db"  
 }  
 // Creating Tables  
 override fun onCreate(db: SQLiteDatabase) {  
 // create notes table  
 db.execSQL(NotesData.CREATE\_TABLE)  
 }  
 // Upgrading database  
 override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {  
 // Drop older table if existed  
 db.execSQL("DROP TABLE IF EXISTS " + NotesData.TABLE\_NAME)  
 // Create tables again  
 onCreate(db)  
 }  
 fun insertNote(note: Note): Long {  
 // get writable database as we want to write data  
 val db = this.*writableDatabase* // insert row  
 val id = db.insert(NotesData.TABLE\_NAME, null, getValues(note))  
 // close db connection  
 db.close()  
 // return newly inserted row id  
 return id  
 }  
 private fun getValues(note: Note): ContentValues {  
 val values = ContentValues()  
 // `id` will be inserted automatically.  
 // no need to add them  
 values.put(NotesData.COLUMN\_NOTE\_TITLE, note.title)  
 values.put(NotesData.COLUMN\_NOTE\_SUB\_TITLE, note.subTitle)  
 values.put(NotesData.COLUMN\_NOTE\_DESCRIPTION, note.Description)  
 values.put(NotesData.COLUMN\_NOTE\_REMINDER\_TIME, note.remindertime)  
 values.put(NotesData.COLUMN\_NOTE\_SET\_REMINDER, note.isReminder)  
 values.put(NotesData.COLUMN\_TIMESTAMP, note.modifiedTime)  
 return values  
 }  
 fun getNote(id: Long): Note {  
 // get readable database as we are not inserting anything  
 val db = this.*readableDatabase* val cursor = db.query(  
 NotesData.TABLE\_NAME,  
 *arrayOf*(NotesData.COLUMN\_ID,  
 NotesData.COLUMN\_NOTE\_TITLE,  
 NotesData.COLUMN\_NOTE\_SUB\_TITLE,  
 NotesData.COLUMN\_NOTE\_DESCRIPTION,  
 NotesData.COLUMN\_NOTE\_SET\_REMINDER,  
 NotesData.COLUMN\_NOTE\_REMINDER\_TIME,  
 NotesData.COLUMN\_TIMESTAMP),  
 NotesData.COLUMN\_ID.toString() + "=?",  
 *arrayOf*(id.toString()),  
 null,  
 null,  
 null,  
 null  
 )  
 cursor?.moveToFirst()  
 val note = getNote(cursor)  
  
 // close the db connection  
 cursor!!.close()  
 return note  
 }  
 private fun getNote(cursor: Cursor): Note  
 {  
 // prepare note object  
 val note = Note(  
 cursor.getString(cursor.getColumnIndexOrThrow(NotesData.COLUMN\_NOTE\_TITLE)),  
 cursor.getString(cursor.getColumnIndexOrThrow(NotesData.COLUMN\_NOTE\_SUB\_TITLE)),  
 cursor.getString(cursor.getColumnIndexOrThrow(NotesData.COLUMN\_NOTE\_DESCRIPTION)),  
 cursor.getString(cursor.getColumnIndexOrThrow(NotesData.COLUMN\_TIMESTAMP))  
 )  
 note.isReminder = (cursor.getInt(cursor.getColumnIndexOrThrow(NotesData.COLUMN\_NOTE\_SET\_REMINDER))==1)  
 note.remindertime = cursor.getLong(cursor.getColumnIndexOrThrow(NotesData.COLUMN\_NOTE\_REMINDER\_TIME))  
 note.id = cursor.getInt(cursor.getColumnIndexOrThrow(NotesData.COLUMN\_ID))  
 return note  
 }  
 // Select All Query  
 val allNotes: ArrayList<Note>  
 get() {  
 val notes = ArrayList<Note>()  
 // Select All Query  
 val selectQuery = "SELECT \* FROM " + NotesData.TABLE\_NAME.toString() + " ORDER BY " +  
 NotesData.COLUMN\_TIMESTAMP.toString() + " DESC"  
 val db = this.*writableDatabase* val cursor = db.rawQuery(selectQuery, null)  
 // looping through all rows and adding to list  
 if (cursor.moveToFirst()) {  
 do {  
 notes.add(getNote(cursor))  
 } while (cursor.moveToNext())  
 }  
 // close db connection  
 db.close()  
 // return notes list  
 return notes  
 }  
 val notesCount: Int  
 get() {  
 val countQuery = "SELECT \* FROM " + NotesData.TABLE\_NAME  
 val db = this.*readableDatabase* val cursor = db.rawQuery(countQuery, null)  
 val count = cursor.*count* cursor.close()  
 // return count  
 return count  
 }  
 fun updateNote(note: Note): Int {  
 val db = this.*writableDatabase* // updating row  
 return db.update(  
 NotesData.TABLE\_NAME,  
 getValues(note),  
 NotesData.COLUMN\_ID + " = ?",  
 *arrayOf*(note.id.toString())  
 )  
 }  
 fun deleteNote(note: Note) {  
 val db = this.*writableDatabase* db.delete(  
 NotesData.TABLE\_NAME,  
 NotesData.COLUMN\_ID + " = ?",  
 *arrayOf*(note.id.toString())  
 )  
 db.close()  
 }  
}

**NotesAdapter.kt:**

package com.example.mad\_20012021033\_p\_11  
  
import android.content.Context  
import android.content.Intent  
import android.view.LayoutInflater  
import android.view.View  
import android.view.ViewGroup  
import androidx.recyclerview.widget.RecyclerView  
import com.example.mad\_20012021033\_p\_11.databinding.NoteItemViewBinding  
  
import java.io.Serializable  
  
class NotesAdapter (private val context: Context, private val array:ArrayList<Note>):  
 RecyclerView.Adapter<NotesAdapter.NotesViewHolder>() {  
 inner class NotesViewHolder(val binding: NoteItemViewBinding): RecyclerView.ViewHolder(binding.*root*)  
  
 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): NotesViewHolder {  
 val binding = NoteItemViewBinding.inflate(LayoutInflater.from(parent.*context*), parent, false)  
 return NotesViewHolder(binding)  
 }  
  
 override fun onBindViewHolder(holder: NotesViewHolder, position: Int) {  
 *with*(holder)**{** *with*(array[position])**{** binding.noteTitle.*text* = this.title  
 binding.noteSubtitle.*text* = this.subTitle  
 binding.noteContent.*text* = this.Description  
 binding.noteDate.*text* = this.modifiedTime  
 val obj = this as Serializable  
 this.calcReminder()  
 if(this.isReminder)  
 {  
 binding.noteReminderDateTime.*visibility* = View.*VISIBLE* binding.noteReminderDateTime.*text* = this.getReminderText()  
 }  
 else  
 binding.noteReminderDateTime.*visibility* = View.*GONE* binding.imgDelete.setOnClickListener**{** (context as MainActivity).deleteNote(position)  
 **}** binding.cardNote.setOnClickListener **{** Intent(this@NotesAdapter.context, NoteViewActivity::class.*java*).*apply* **{** putExtra("Object",obj)  
 this@NotesAdapter.context.startActivity(this)  
 **}  
 }** binding.imgEdit.setOnClickListener **{** (context as MainActivity).showAlertDialog(  
 NoteMode.*edit*,  
 "Edit Note",  
 this,  
 position,  
 this@NotesAdapter  
 )  
 **}  
 }  
 }** }  
  
 override fun getItemCount(): Int {  
 return array.size  
 }  
}

**NotesData.kt:**

package com.example.mad\_20012021033\_p\_11  
  
class NotesData {  
 companion object{  
 const val TABLE\_NAME = "notes"  
  
 const val COLUMN\_ID = "id"  
 const val COLUMN\_NOTE\_TITLE = "note\_title"  
 const val COLUMN\_NOTE\_SUB\_TITLE = "note\_sub\_title"  
 const val COLUMN\_NOTE\_DESCRIPTION = "note\_description"  
 const val COLUMN\_NOTE\_SET\_REMINDER = "note\_set\_reminder"  
 const val COLUMN\_NOTE\_REMINDER\_TIME = "note\_set\_reminder\_time"  
 const val COLUMN\_TIMESTAMP = "note\_modified\_timestamp"  
  
 // Create table SQL query  
 val CREATE\_TABLE = ("CREATE TABLE " + TABLE\_NAME + "("  
 + COLUMN\_ID + " INTEGER PRIMARY KEY AUTOINCREMENT,"  
 + COLUMN\_NOTE\_TITLE + " TEXT,"  
 + COLUMN\_NOTE\_SUB\_TITLE + " TEXT,"  
 + COLUMN\_NOTE\_DESCRIPTION + " TEXT,"  
 + COLUMN\_NOTE\_SET\_REMINDER + " INTEGER,"  
 + COLUMN\_NOTE\_REMINDER\_TIME + " INTEGER,"  
 + COLUMN\_TIMESTAMP + " TEXT"  
 + ")")  
 }  
}

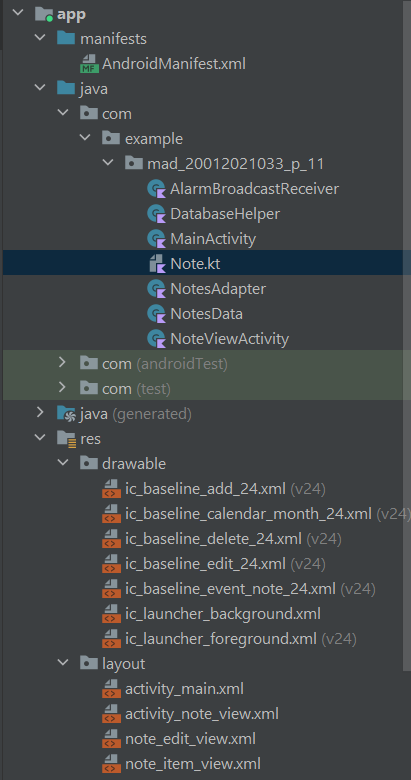
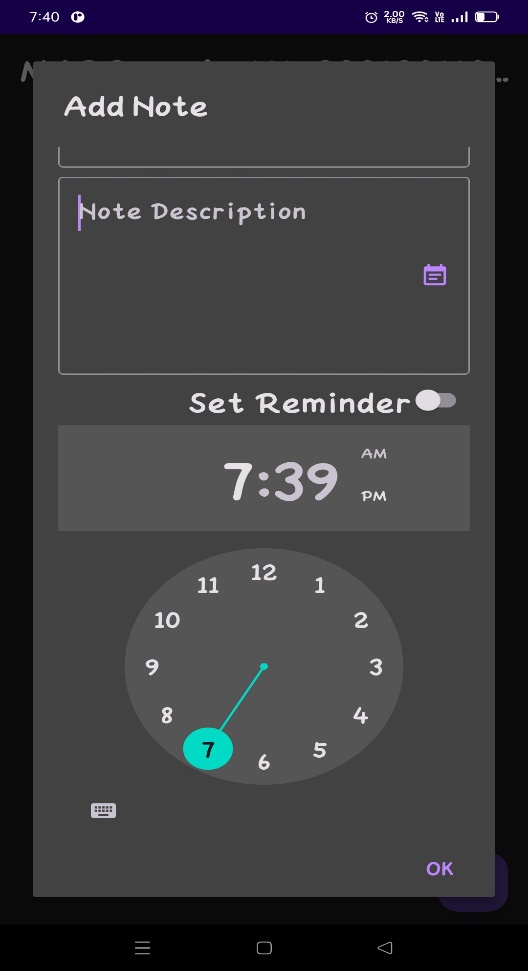
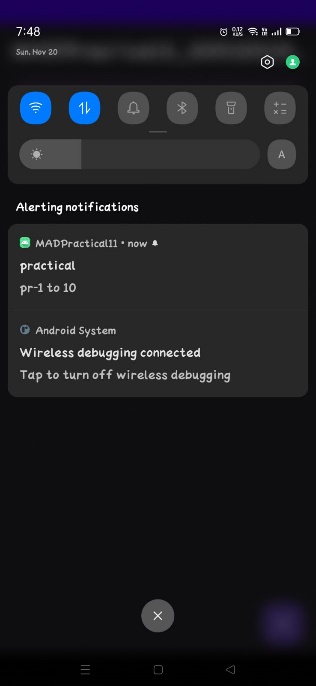
**NoteViewActivity.kt:**

package com.example.mad\_20012021033\_p\_11  
  
import androidx.appcompat.app.AppCompatActivity  
import android.os.Bundle  
import android.view.View  
import com.example.mad\_20012021033\_p\_11.databinding.ActivityNoteViewBinding  
  
  
class NoteViewActivity : AppCompatActivity() {  
 private lateinit var binding: ActivityNoteViewBinding  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
// setContentView(R.layout.activity\_note\_view)  
 binding = ActivityNoteViewBinding.inflate(*layoutInflater*)  
 setContentView(binding.*root*)  
 setSupportActionBar(binding.toolbar)  
 var note = *intent*.getSerializableExtra("Object") as Note  
 *with*(note)  
 **{** binding.noteTitle.*text* = this.title  
 binding.noteSubtitle.*text* = this.subTitle  
 binding.noteContent.*text* = this.Description  
 binding.noteDate.*text* = this.modifiedTime  
 this.calcReminder()  
 if (this.isReminder){  
 binding.noteReminderDateTime.*visibility* = View.*VISIBLE* binding.noteReminderDateTime.*text* = this.getReminderText()  
 }  
 else{  
 binding.noteReminderDateTime.*visibility* = View.*GONE* }  
 **}** }  
}

**Note.kt:**

package com.example.mad\_20012021033\_p\_11  
  
import android.app.AlarmManager  
import android.app.PendingIntent  
import android.content.Context  
import android.content.Context.*ALARM\_SERVICE*import android.content.Intent  
import java.io.Serializable  
import java.text.DateFormat  
import java.text.SimpleDateFormat  
import java.util.\*  
enum class NoteMode{  
 *edit*,  
 *add*}  
class Note(var title:String, var subTitle:String, var Description:String, var modifiedTime:String, var isReminder: Boolean = false):Serializable  
{  
 var remindertime:Long = System.currentTimeMillis()  
 var id = noteIdGeneration()  
 set(value) {  
 **field** = value  
 if(idNote < value)  
 idNote = value  
 }  
  
 constructor():this("","","","")  
 { }  
 constructor(note: Note) : this(note.title,note.subTitle,note.Description,note.modifiedTime,note.isReminder) {  
 remindertime = note.remindertime  
 }  
  
 fun isValid() :Boolean{  
 if(title.*isEmpty*() || Description.*isEmpty*())  
 return false  
 return true  
 }  
 fun changeValue(newValue: Note)  
 {  
 title = newValue.title  
 subTitle = newValue.subTitle  
 Description = newValue.Description  
 modifiedTime = newValue.modifiedTime  
 isReminder = newValue.isReminder  
 remindertime = newValue.remindertime  
 }  
 fun getReminderText() :String  
 {  
 return "Reminder: "+(SimpleDateFormat("MMM, dd yyyy hh:mm a") as DateFormat).format(  
 Date(remindertime)  
 )  
 }  
 fun saveNote(context: Context)  
 {  
 if(isReminder)  
 {  
 setReminder(context,this)  
 }  
 }  
 fun getHour():Int{  
 val cal = Calendar.getInstance()  
 cal.*time* = Date(remindertime)  
 return cal[Calendar.*HOUR\_OF\_DAY*]  
 }  
 fun getMinute():Int{  
 val cal = Calendar.getInstance()  
 cal.*time* = Date(remindertime)  
 return cal[Calendar.*MINUTE*]  
 }  
 fun calcReminder()  
 {  
 if(remindertime < System.currentTimeMillis())  
 isReminder = false  
 }  
  
 override fun toString(): String {  
 return "$id\n"+title +"\n"+subTitle +"\n"+Description+"\nReminder:$isReminder" +"\n"+getReminderText()  
 }  
  
  
 companion object {  
 var idNote = 0  
 fun noteIdGeneration():Int  
 {  
 idNote++  
 return idNote  
 }  
 val REMINDER\_REQUEST\_CODE = 1000  
 val NOTE\_ID\_KEY = "Id"  
 val NOTE\_TITLE\_KEY = "Title"  
 val NOTE\_SUBTITLE\_KEY = "SubTitle"  
 val NOTE\_DESCRIPTION\_KEY = "Description"  
 val NOTE\_MODIFIED\_TIME\_KEY = "ModifiedTime"  
 val NOTE\_REMINDER\_TIME\_KEY = "ReminderTime"  
  
 fun getCurrentDateTime(): String {  
 val cal = Calendar.getInstance()  
 val df: DateFormat = SimpleDateFormat("MMM, dd yyyy hh:mm:ss a")  
 return df.format(cal.*time*)  
 }  
 fun getMillis(hour:Int,min:Int):Long  
 {  
 val setcalendar = Calendar.getInstance()  
 setcalendar[Calendar.*HOUR\_OF\_DAY*] = hour  
 setcalendar[Calendar.*MINUTE*] = min  
 setcalendar[Calendar.*SECOND*] = 0  
 return setcalendar.*timeInMillis* }  
  
 fun setReminder(context: Context, note: Note) {  
 val intent = Intent(context, AlarmBroadcastReceiver::class.*java*)  
 intent.putExtra(NOTE\_ID\_KEY, note.id)  
 intent.putExtra(NOTE\_TITLE\_KEY, note.title)  
 intent.putExtra(NOTE\_SUBTITLE\_KEY, note.subTitle)  
 intent.putExtra(NOTE\_DESCRIPTION\_KEY, note.Description)  
 intent.putExtra(NOTE\_MODIFIED\_TIME\_KEY, note.modifiedTime)  
 intent.putExtra(NOTE\_REMINDER\_TIME\_KEY, note.remindertime)  
  
 val pendingIntent =  
 PendingIntent.getBroadcast(context, note.id, intent, PendingIntent.*FLAG\_CANCEL\_CURRENT*)  
 val alarmManager = context.getSystemService(*ALARM\_SERVICE*) as AlarmManager  
  
 if(note.isReminder) {  
 alarmManager.setExact(  
 AlarmManager.*RTC\_WAKEUP*,  
 note.remindertime,  
 pendingIntent  
 )  
 }  
 else  
 alarmManager.cancel(pendingIntent)  
 }  
 }  
}

**Output:**

**  **