Mobile Device Programming

Final Assignment

Created by

MD FAHIM MUNTASIR (王一然)

ID: 1911521213

Class: 19lq CST

Date: 1st February, 2023

Complete and submit an Android program and related report.

The requirements are as follows:

- No less than 5 pages (including activity or fragment)
- Use MVVM pattern
- Use database to store data

Activity or Fragment:

An activity is one screen of an app. In that way the activity is very similar to a window in the Windows operating system.

The most specific block of the user interface is the activity. An Android app contains activities, meaning one or more screens.

Examples: Login screen, sign up screen, and home screen.

An activity class is per definition a class in Android. Every application which has UI must inherit it to create a window. An activity in Android is a specific combination of XML files and JAVA files. It is basically a container that contains the design as well as coding stuff. XML files provide the design of the screen and JAVA files deal with all coding stuff like handles, what is happening, design files, etc. JAVA files and XML files make your activity complete.

An activity goes through a number of states. Activity lifecycle in Android manages the different states of activity, such as when the activity starts or stops, etc. All of these states are managed by callback methods.

There are seven callback methods:

onCreate(): This callback method must be created by you, this is fired when your activity is created by the system. For example, when you open the app it will call onCreate(), onStart(), and onResume() methods. These three methods are initiated when you open an app.

onStart(): There should always be an onStart() callback after onCreate() finishes. In the started state users will be able to see the activity but they are not ready for user interaction.

onResume(): In this, activities will be in the foreground and ready for user interaction. You can understand the use of this callback from this example: “when the user presses the phone’s answer button it will give onPause() after you end up calling it will again give onResume()”.

onPause(): This callback occurs when there is a case of another activity starting in front of the current activity. It is the opposite of onResume(). An example for this callback can be “when you open the app, that is another app from the notification bar, or open settings then it will call onPause() and onStop()”.

onStop(): It is the opposite of onStart(). In this case, activity will no longer be visible to the users.

onRestart(): This occurs when the activity is stopped and before the activity starts again. It is a less common callback.

onDestroy(): It is opposite of onCreate(). This starts when the system needs to free memory or when finish() is called. It is used for cleanup which is a very common activity.

On the other hand**s Android Fragment** is the part of activity, it is also known as sub-activity. There can be more than one fragment in an activity. Fragments represent multiple screens inside one activity. Android fragment lifecycle is affected by activity lifecycle because fragments are included in activity. Each fragment has its own life cycle methods that is affected by activity life cycle because fragments are embedded in activity. The **FragmentManager** class is responsible to make interaction between fragment objects.

The lifecycle of android fragment is like the activity lifecycle. There are 12 lifecycle methods for fragment.

onAttach(Activity): It is called only once when it is attached with activity.

onCreate(Bundle): It is used to initialize the fragment.

onCreateView(LayoutInflater, ViewGroup, Bundle): Creates and returns view hierarchy.

onActivityCreated(Bundle): It is invoked after the completion of onCreate() method.

onViewStateRestored(Bundle): It provides information to the fragment that all the saved state of fragment view hierarchy has been restored.

onStart(): Makes the fragment visible.

onResume(): Makes the fragment interactive.

onPause(): It is called when fragment is no longer interactive.

onStop(): It is called when fragment is no longer visible.

onDestroyView(): It is allows the fragment to clean up resources.

onDestroy(): It is allows the fragment to do final clean-up of fragment state.

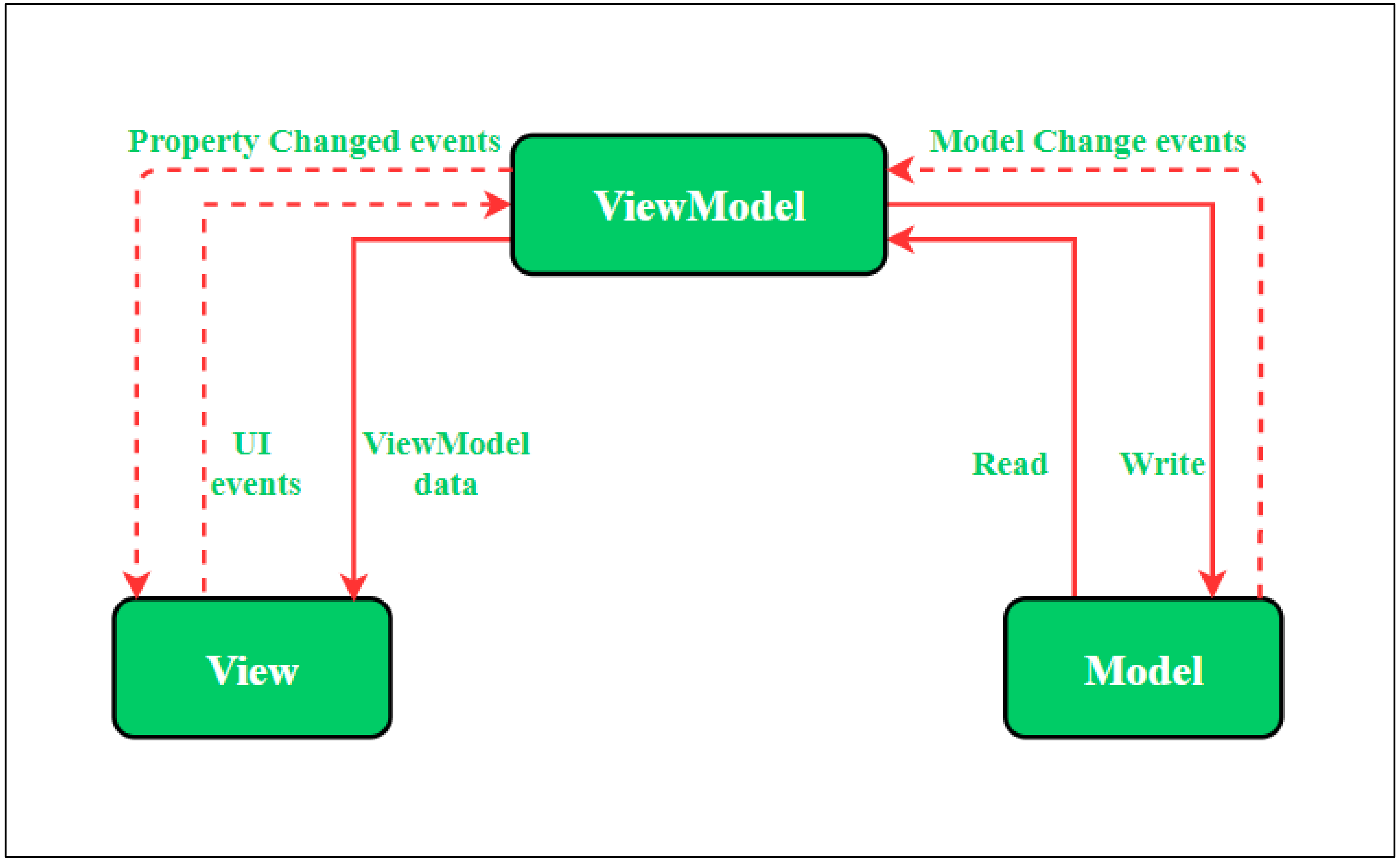
onDetach(): It is called immediately prior to the fragment no longer being associated with its activity.

MVVM Pattern:

Model-View-ViewModel (MVVM) is the industry-recognized software architecture pattern that overcomes all drawbacks of MVP and MVC design patterns. MVVM suggests separating the data presentation logic (Views or UI) from the core business logic part of the application.

The separate code layers of MVVM are:

- **Model:** This layer is responsible for the abstraction of the data sources. Model and ViewModel work together to get and save the data.
- **View:** The purpose of this layer is to inform the ViewModel about the user’s action. This layer observes the ViewModel and does not contain any kind of application logic.
- **ViewModel:** It exposes those data streams which are relevant to the View. Moreover, it serves as a link between the Model and the View.


Property Changed events
ViewModel
Model Change events
UI events
ViewModel data
Read
Write
View
Model

MVVM

MVVM pattern has some similarities with the MVP (Model-View-Presenter) design pattern as the Presenter role is played by ViewModel. However, the drawbacks of the MVP pattern has been solved by MVVM in the following ways:

1. ViewModel does not hold any kind of reference to the View.
2. Many to-1 relationships exist between View and ViewModel.
3. No triggering methods to update the View.

Database to store:

A database for Android is a form of persistent data storage intended for use in apps for Android devices. It often consists of on-device, local storage so the app continues to be available even if the device loses connectivity. Due to its inclusion in the Android Software Development Kit (SDK), SQLite, an open-source relational database, is the most common database technology associated with Android applications. For Android apps, SQLite is most often paired with Room, a framework for managing the lifecycle of objects. However, as this article will go on to discuss, there are other simpler options for modern mobile applications.

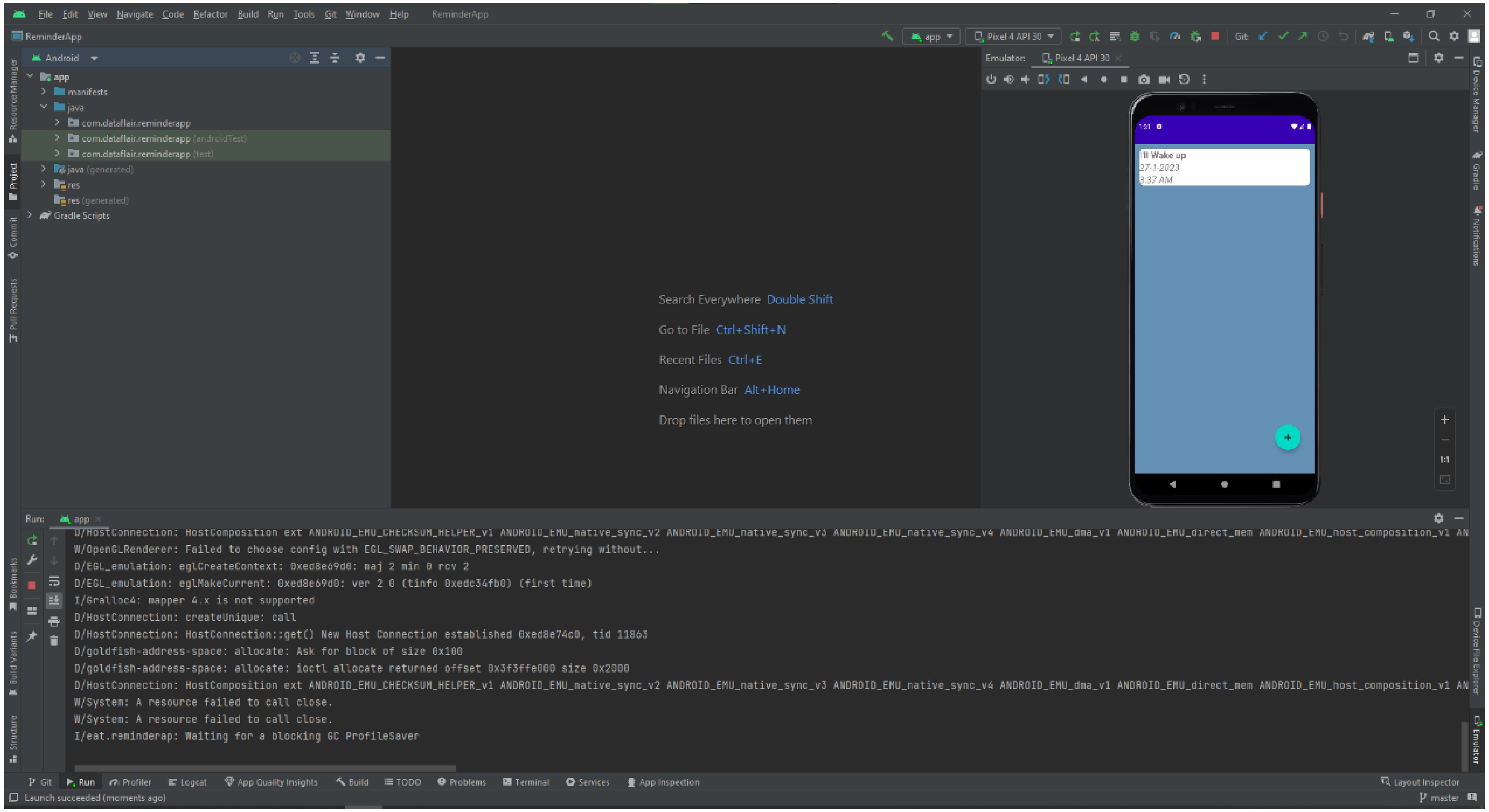
Different types of Android database

Relational database: Uses tables and shared columns, aka keys, to form relationships between tables. Most common example is SQLite, available inside the Android SDK. Great if relationships between entities are important.

Key-Value database: Uses a key to single value mapping to store information. Often used in Android databases for saving things such as user settings. Example would be Shared Preferences for Android.

Object-oriented database: Can work with complex data objects as seen in object-oriented programming languages. Able to do fast queries with complex data. Code is often simpler as well due to closeness in structure between the database and objects in code. Realm is an object-oriented database.

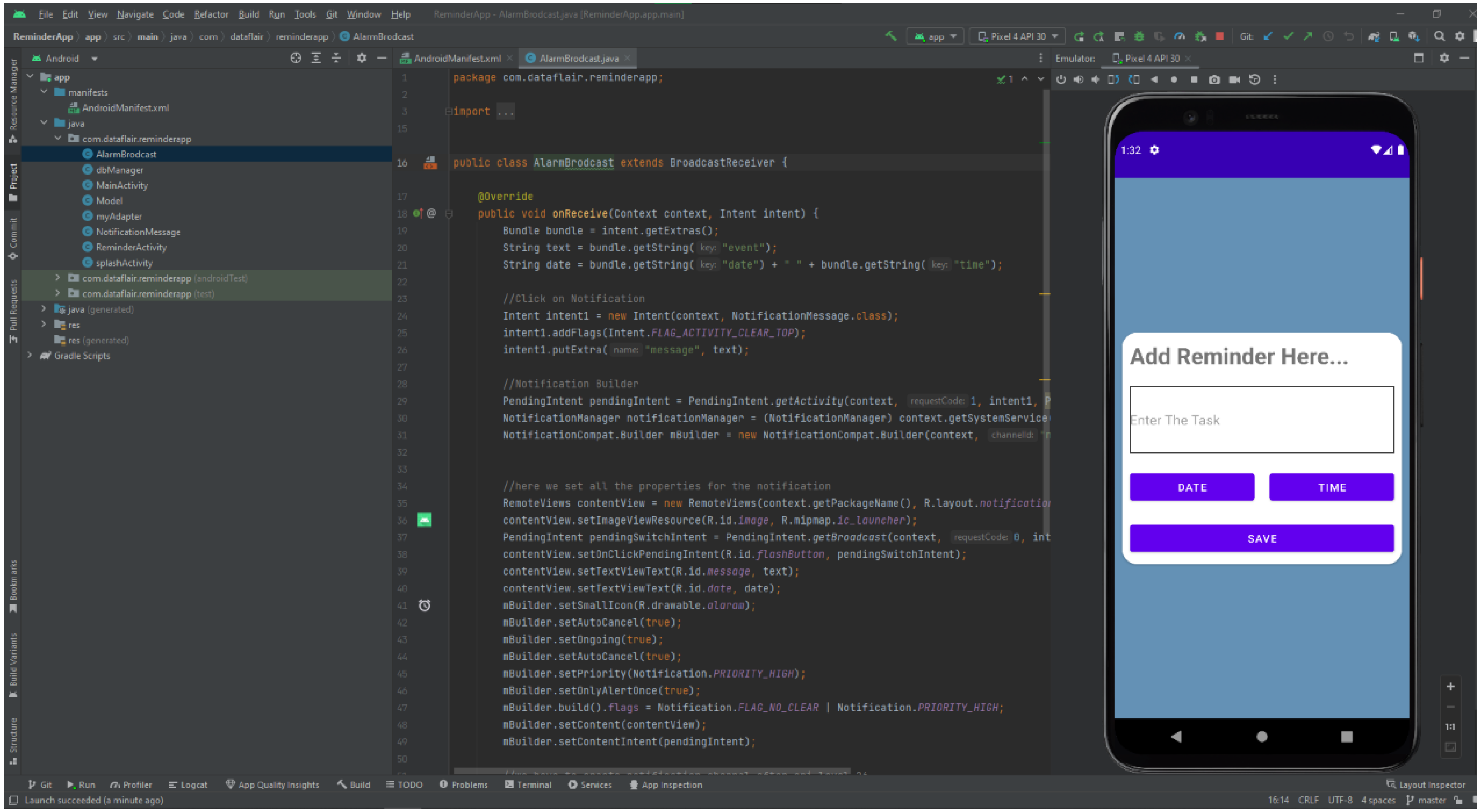
Android Program


File Edit View Navigate Code Refactor Build Run Tools Git Window Help ReminderApp
ReminderApp
app
Pixel 4 API 30
Android
app
manifests
java
com.dataflair.reminderapp
com.dataflair.reminderapp (androidTest)
com.dataflair.reminderapp (test)
java (generated)
res
res (generated)
Gradle Scripts
Search Everywhere Double Shift
Go to File Ctrl+Shift+N
Recent Files Ctrl+E
Navigation Bar Alt+Home
Drop files here to open them
Emulator: Pixel 4 API 30
Run: app
Git Run Profiler Logcat App Quality Insights Build TODO Problems Terminal Services App Inspection
Layout Inspector
master

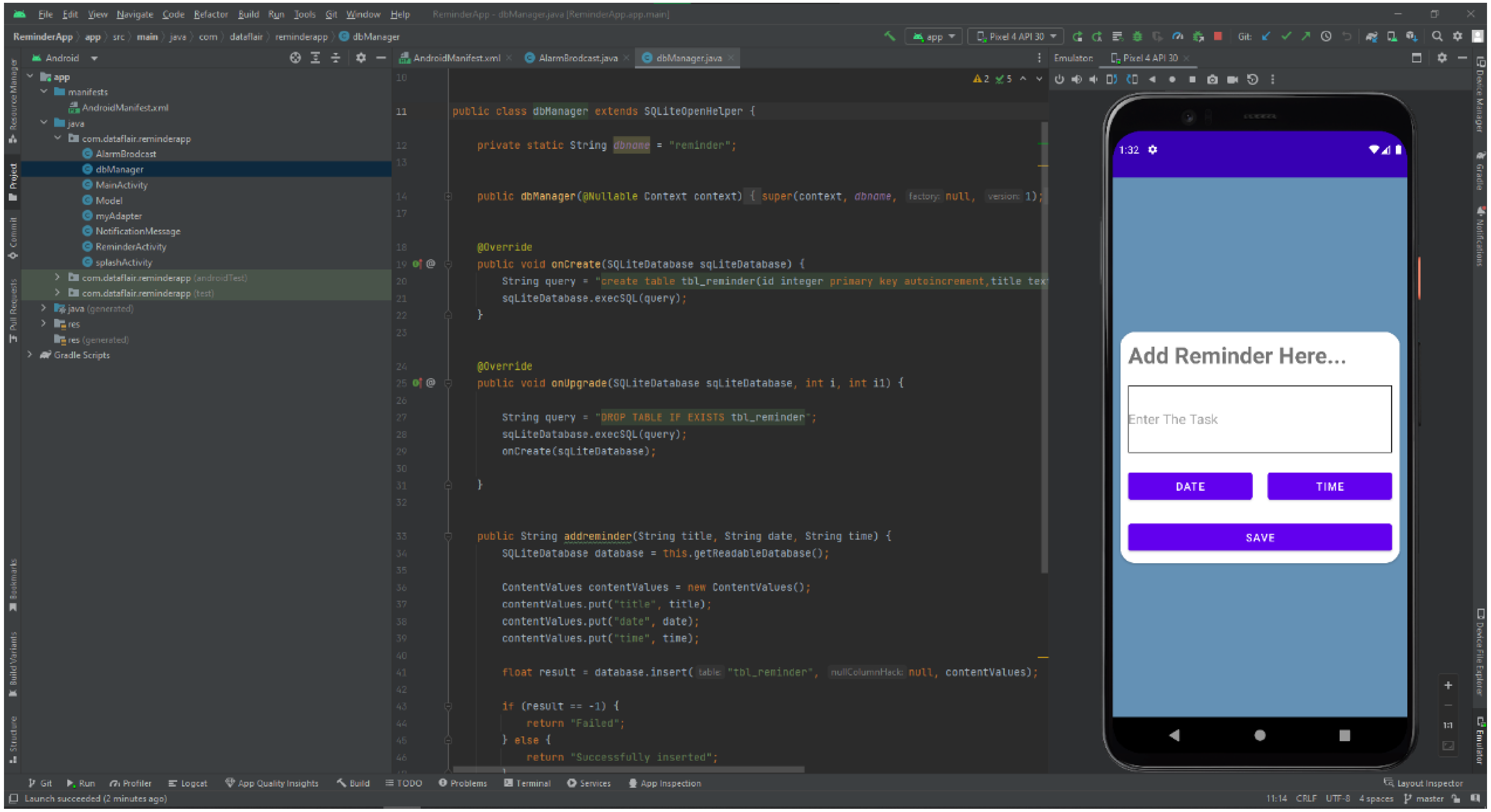
I'll Wake up
27-1-2023
3:37 AM

```
D/HostConnection: HostComposition ext ANDROID_EMU_CHECKSUM_HELPER_v1 ANDROID_EMU_native_sync_v2 ANDROID_EMU_native_sync_v3 ANDROID_EMU_native_sync_v4 ANDROID_EMU_dma_v1 ANDROID_EMU_direct_mem ANDROID_EMU_host_composition_v1 AN
W/OpenGLRenderer: Failed to choose config with EGL_SWAP_BEHAVIOR_PRESERVED, retrying without...
D/EGL_emulation: eglCreateContext: 0xed8e69d0: maj 2 min 0 rcv 2
D/EGL_emulation: eglMakeCurrent: 0xed8e69d0: ver 2 0 (tinfo 0xedc34fb0) (first time)
I/Gralloc4: mapper 4.x is not supported
D/HostConnection: createUnique: call
D/HostConnection: HostConnection::get() New Host Connection established 0xed8e74c0, tid 11863
D/goldfish-address-space: allocate: Ask for block of size 0x100
D/goldfish-address-space: allocate: ioctl allocate returned offset 0x3f3ffe000 size 0x2000
D/HostConnection: HostComposition ext ANDROID_EMU_CHECKSUM_HELPER_v1 ANDROID_EMU_native_sync_v2 ANDROID_EMU_native_sync_v3 ANDROID_EMU_native_sync_v4 ANDROID_EMU_dma_v1 ANDROID_EMU_direct_mem ANDROID_EMU_host_composition_v1 AN
W/System: A resource failed to call close.
W/System: A resource failed to call close.
I/eat.reminderap: Waiting for a blocking GC ProfileSaver
```

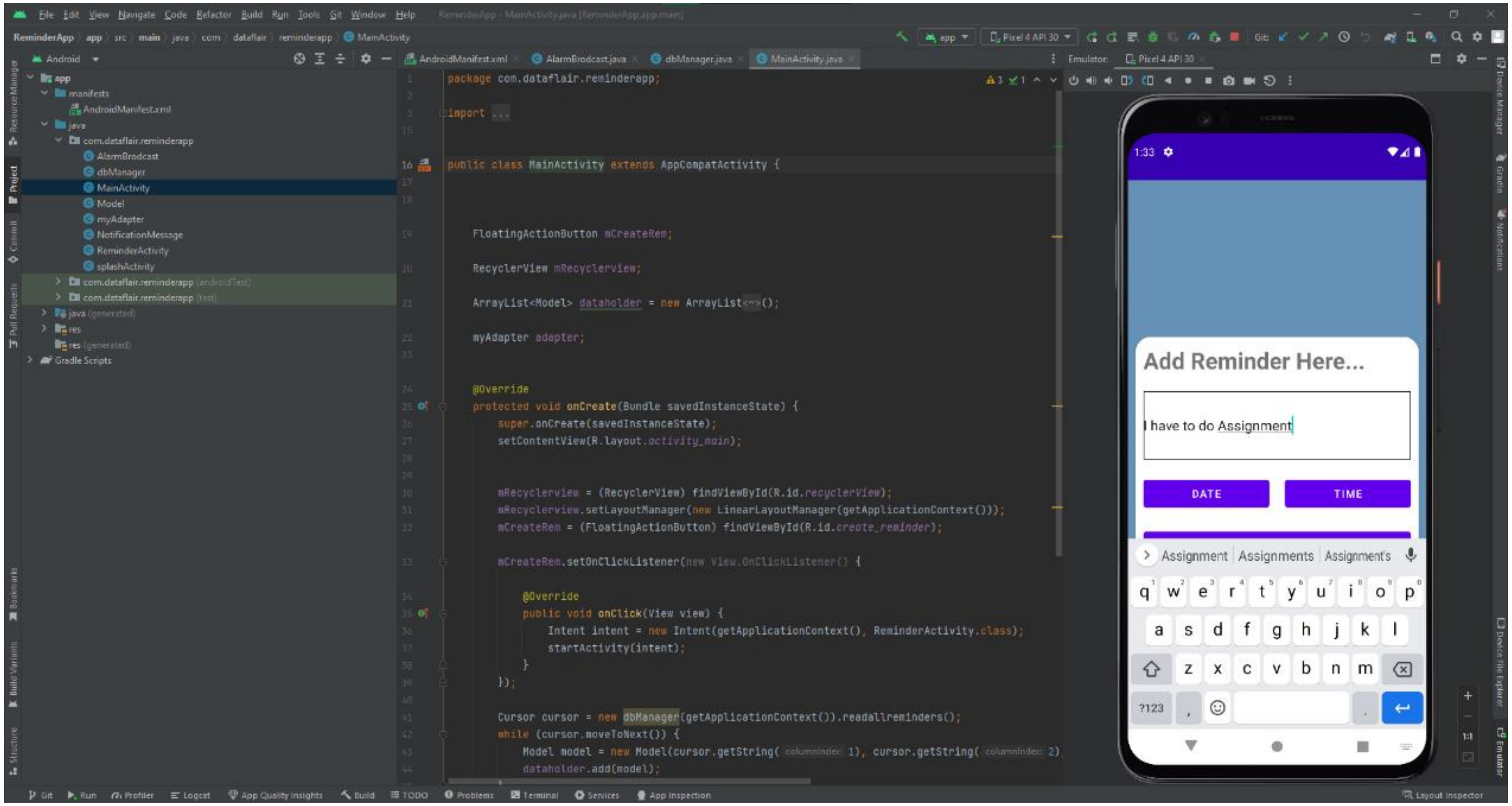
Launch succeeded (moments ago)



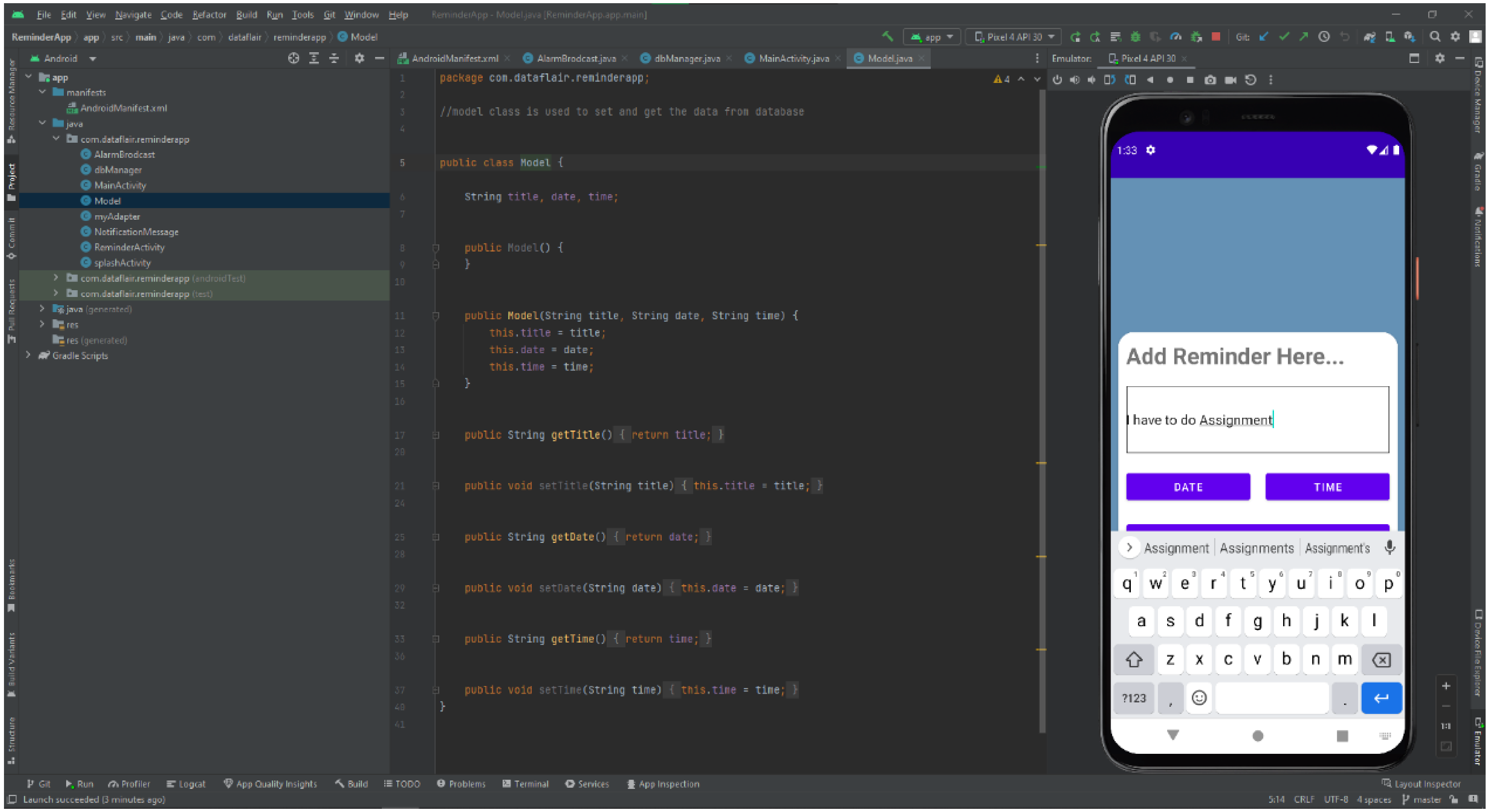
```java
package com.dataflair.reminderapp;

import ...

public class AlarmBrodcast extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {
        Bundle bundle = intent.getExtras();
        String text = bundle.getString( key: "event");
        String date = bundle.getString( key: "date") + " " + bundle.getString( key: "time");

        //Click on Notification
        Intent intent1 = new Intent(context, NotificationMessage.class);
        intent1.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        intent1.putExtra( name: "message", text);

        //Notification Builder
        PendingIntent pendingIntent = PendingIntent.getActivity(context, requestCode: 1, intent1,
        NotificationManager notificationManager = (NotificationManager) context.getSystemService
        NotificationCompat.Builder mBuilder = new NotificationCompat.Builder(context, channelId:


        //here we set all the properties for the notification
        RemoteViews contentView = new RemoteViews(context.getPackageName(), R.layout.notificatio
        contentView.setImageViewResource(R.id.image, R.mipmap.ic_launcher);
        PendingIntent pendingSwitchIntent = PendingIntent.getBroadcast(context, requestCode: 0, int
        contentView.setOnClickPendingIntent(R.id.flashButton, pendingSwitchIntent);
        contentView.setTextViewText(R.id.message, text);
        contentView.setTextViewText(R.id.date, date);
        mBuilder.setSmallIcon(R.drawable.alaram);
        mBuilder.setAutoCancel(true);
        mBuilder.setOngoing(true);
        mBuilder.setAutoCancel(true);
        mBuilder.setPriority(Notification.PRIORITY_HIGH);
        mBuilder.setOnlyAlertOnce(true);
        mBuilder.build().flags = Notification.FLAG_NO_CLEAR | Notification.PRIORITY_HIGH;
        mBuilder.setContent(contentView);
        mBuilder.setContentIntent(pendingIntent);

```



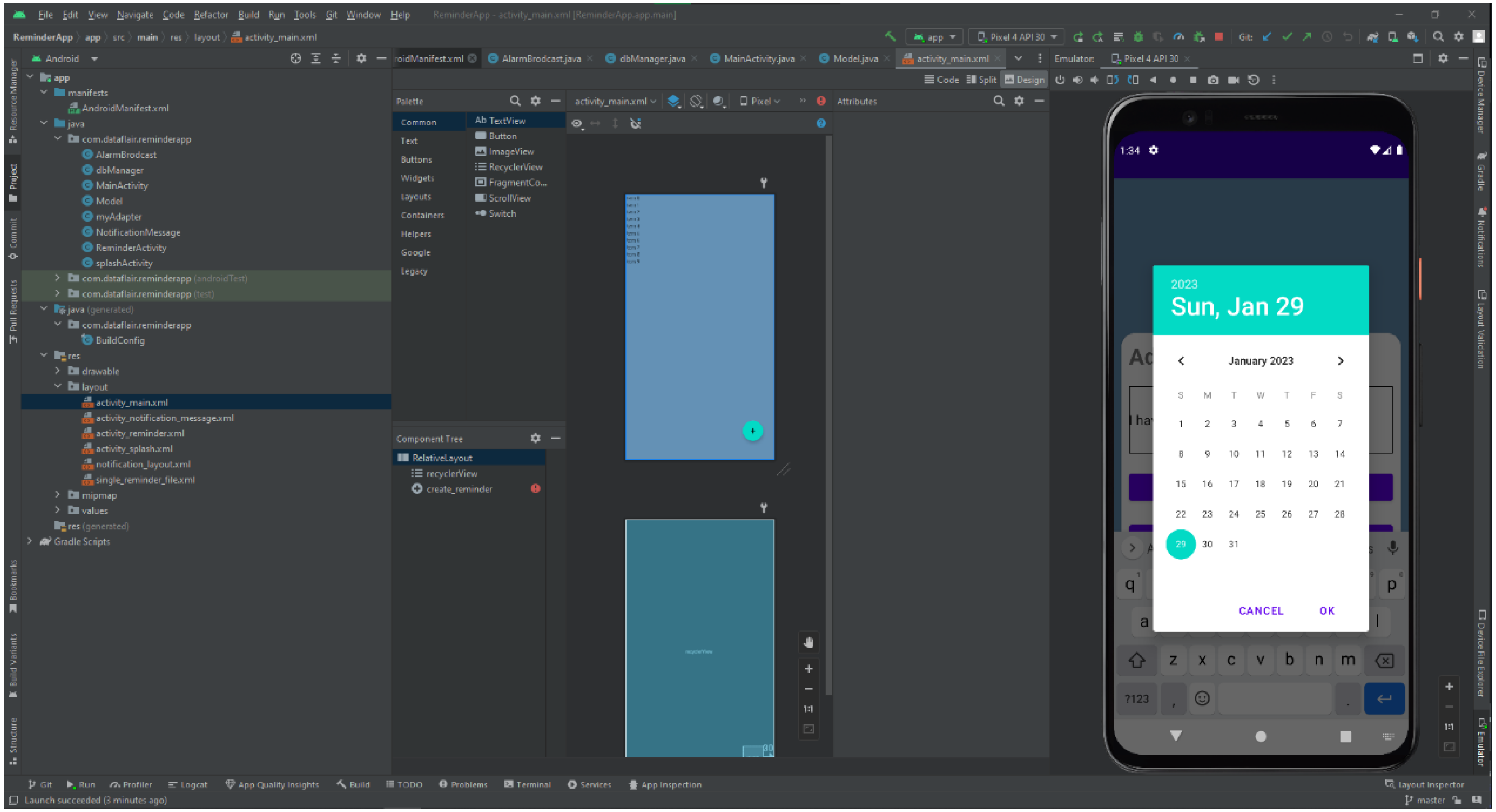
```java
public class dbManager extends SQLiteOpenHelper {

    private static String dbname = "reminder";


    public dbManager(@Nullable Context context) { super(context, dbname, factory: null, version: 1);


    @Override
    public void onCreate(SQLiteDatabase sqLiteDatabase) {
        String query = "create table tbl_reminder(id integer primary key autoincrement,title tex
        sqLiteDatabase.execSQL(query);
    }


    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {

        String query = "DROP TABLE IF EXISTS tbl_reminder";
        sqLiteDatabase.execSQL(query);
        onCreate(sqLiteDatabase);

    }


    public String addreminder(String title, String date, String time) {
        SQLiteDatabase database = this.getReadableDatabase();

        ContentValues contentValues = new ContentValues();
        contentValues.put("title", title);
        contentValues.put("date", date);
        contentValues.put("time", time);

        float result = database.insert( table: "tbl_reminder", nullColumnHack: null, contentValues);

        if (result == -1) {
            return "Failed";
        } else {
            return "Successfully inserted";
```

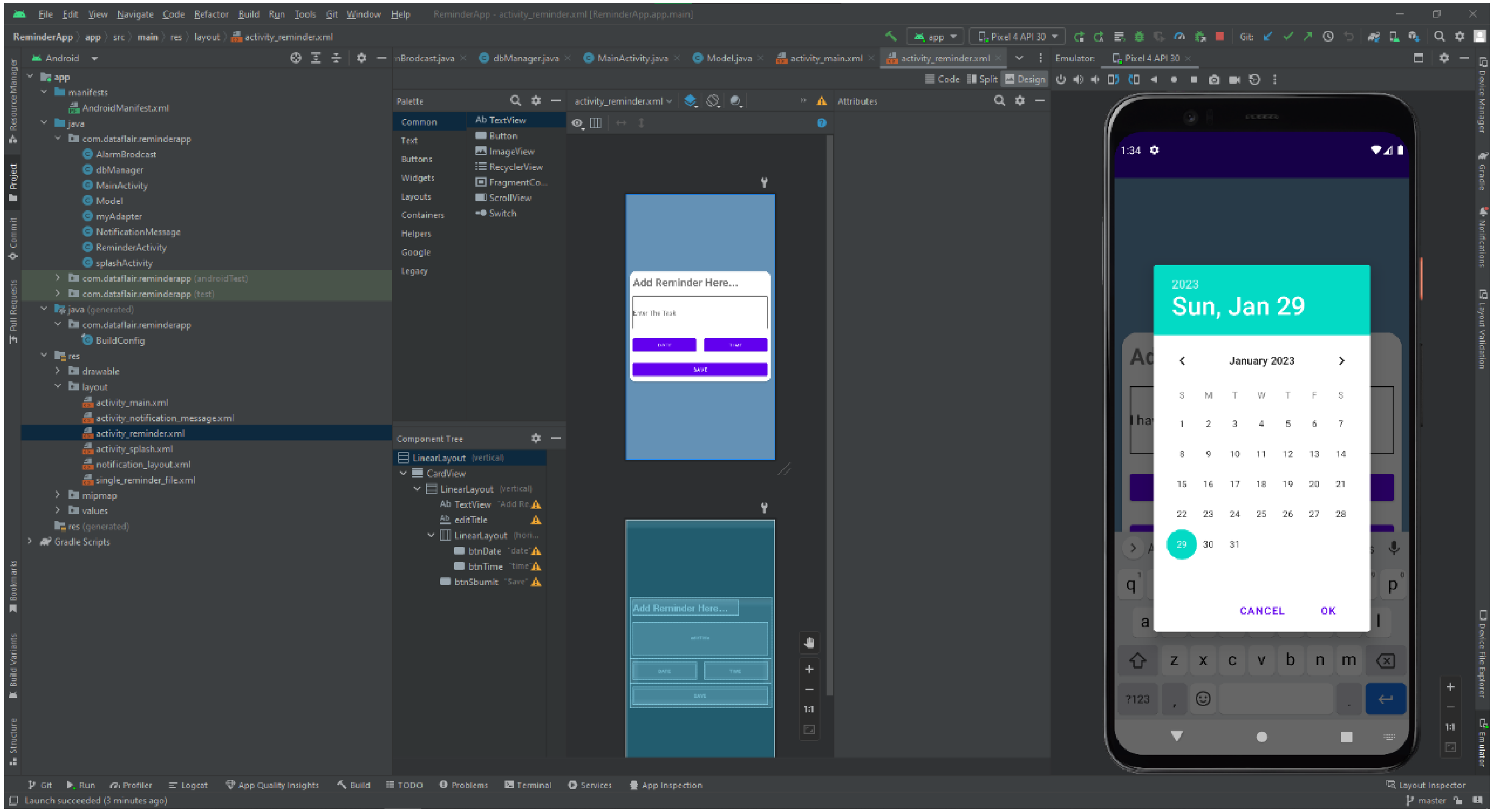


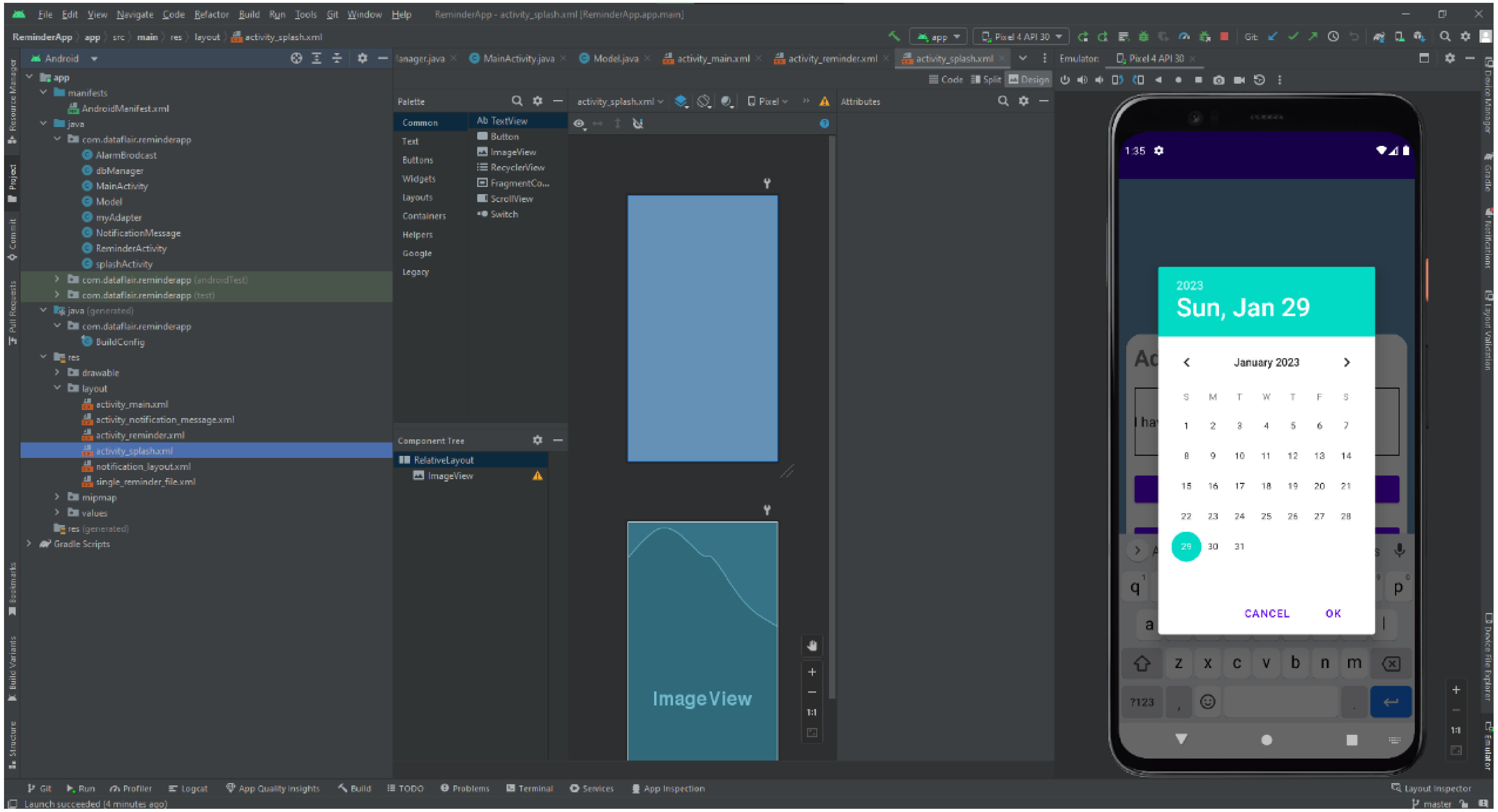
```java
package com.dataflair.reminderapp;

import ...

public class MainActivity extends AppCompatActivity {


    FloatingActionButton mCreateRem;

    RecyclerView mRecyclerview;

    ArrayList<Model> dataholder = new ArrayList<~>();

    myAdapter adapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);


        mRecyclerview = (RecyclerView) findViewById(R.id.recyclerView);
        mRecyclerview.setLayoutManager(new LinearLayoutManager(getApplicationContext()));
        mCreateRem = (FloatingActionButton) findViewById(R.id.create_reminder);

        mCreateRem.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View view) {
                Intent intent = new Intent(getApplicationContext(), ReminderActivity.class);
                startActivity(intent);
            }
        });

        Cursor cursor = new dbManager(getApplicationContext()).readallreminders();
        while (cursor.moveToNext()) {
            Model model = new Model(cursor.getString( columnIndex: 1), cursor.getString( columnIndex: 2)
            dataholder.add(model);
```

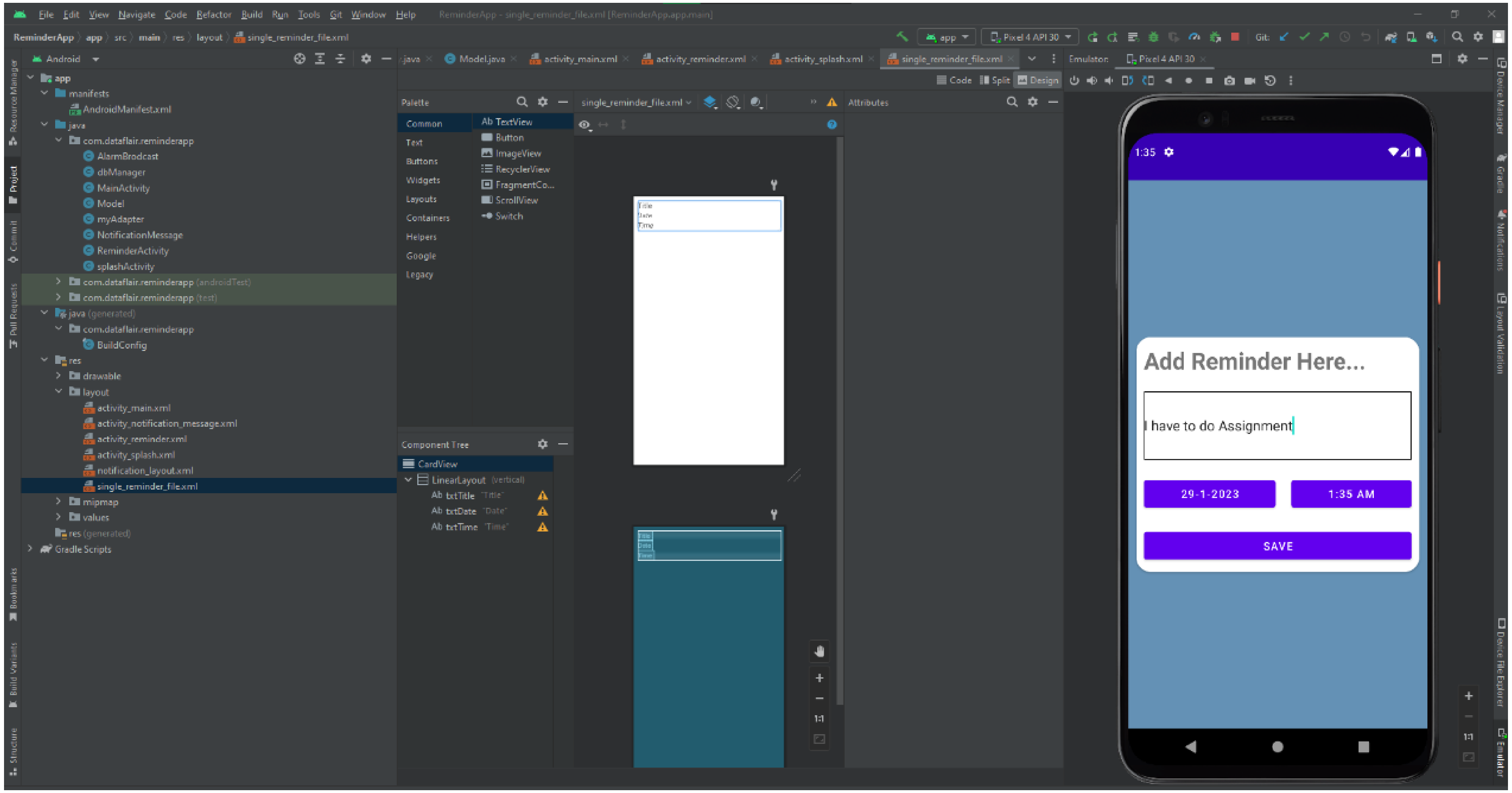


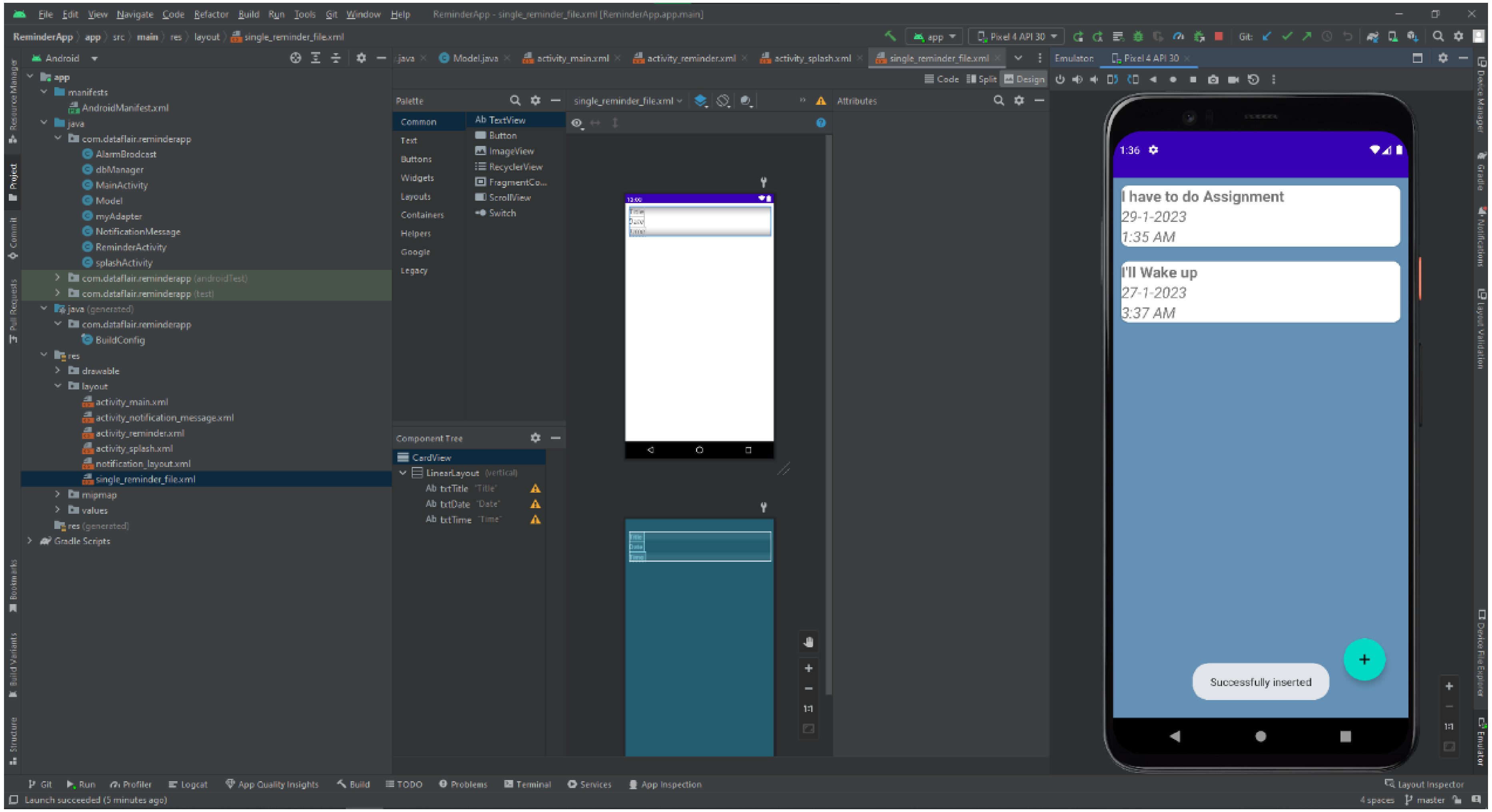
```java
package com.dataflair.reminderapp;

//model class is used to set and get the data from database


public class Model {

    String title, date, time;


    public Model() {
    }


    public Model(String title, String date, String time) {
        this.title = title;
        this.date = date;
        this.time = time;
    }


    public String getTitle() { return title; }


    public void setTitle(String title) { this.title = title; }


    public String getDate() { return date; }


    public void setDate(String date) { this.date = date; }


    public String getTime() { return time; }


    public void setTime(String time) { this.time = time; }
}
```


ReminderApp - activity_main.xml [ReminderApp.app.main]
ReminderApp › app › src › main › res › layout › activity_main.xml
app
manifests
AndroidManifest.xml
java
com.dataflair.reminderapp
AlarmBrodcast
dbManager
MainActivity
Model
myAdapter
NotificationMessage
ReminderActivity
splashActivity
com.dataflair.reminderapp (androidTest)
com.dataflair.reminderapp (test)
java (generated)
BuildConfig
res
drawable
layout
activity_main.xml
activity_notification_message.xml
activity_reminder.xml
activity_splash.xml
notification_layout.xml
single_reminder_file.xml
mipmap
values
res (generated)
Gradle Scripts
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Helpers
Google
Legacy
TextView
Button
ImageView
RecyclerView
FragmentCo...
ScrollView
Switch
Component Tree
RelativeLayout
recyclerView
create_reminder
Attributes
Emulator: Pixel 4 API 30
2023
Sun, Jan 29
January 2023
CANCEL
OK
Launch succeeded (3 minutes ago)


ReminderApp - activity_reminder.xml [ReminderApp.app.main]
ReminderApp › app › src › main › res › layout › activity_reminder.xml
app
manifests
AndroidManifest.xml
java
com.dataflair.reminderapp
AlarmBrodcast
dbManager
MainActivity
Model
myAdapter
NotificationMessage
ReminderActivity
splashActivity
com.dataflair.reminderapp (androidTest)
com.dataflair.reminderapp (test)
java (generated)
BuildConfig
res
drawable
layout
activity_main.xml
activity_notification_message.xml
activity_reminder.xml
activity_splash.xml
notification_layout.xml
single_reminder_file.xml
mipmap
values
res (generated)
Gradle Scripts
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Helpers
Google
Legacy
TextView
Button
ImageView
RecyclerView
FragmentCo...
ScrollView
Switch
Component Tree
LinearLayout (vertical)
CardView
LinearLayout (vertical)
TextView "Add Re...
editTitle
LinearLayout (hori...
btnDate "date"
btnTime "time"
btnSbumit "Save"
Add Reminder Here...
Enter the task
DATE
TIME
SAVE
Emulator: Pixel 4 API 30
1:34
2023
Sun, Jan 29
January 2023
S M T W T F S
CANCEL
OK
Launch succeeded (3 minutes ago)


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
ReminderApp - activity_splash.xml [ReminderApp.app.main]
ReminderApp › app › src › main › res › layout › activity_splash.xml
app
Pixel 4 API 30
Android
app
manifests
AndroidManifest.xml
java
com.dataflair.reminderapp
AlarmBrodcast
dbManager
MainActivity
Model
myAdapter
NotificationMessage
ReminderActivity
splashActivity
com.dataflair.reminderapp (androidTest)
com.dataflair.reminderapp (test)
java (generated)
com.dataflair.reminderapp
BuildConfig
res
drawable
layout
activity_main.xml
activity_notification_message.xml
activity_reminder.xml
activity_splash.xml
notification_layout.xml
single_reminder_file.xml
mipmap
values
res (generated)
Gradle Scripts
lanager.java
MainActivity.java
Model.java
activity_main.xml
activity_reminder.xml
activity_splash.xml
Code
Split
Design
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Helpers
Google
Legacy
Ab TextView
Button
ImageView
RecyclerView
FragmentCo...
ScrollView
Switch
Component Tree
RelativeLayout
ImageView
activity_splash.xml
Pixel
Attributes
ImageView
Emulator:
Pixel 4 API 30
1:35
2023
Sun, Jan 29
January 2023
S M T W T F S
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31
CANCEL
OK
Git
Run
Profiler
Logcat
App Quality Insights
Build
TODO
Problems
Terminal
Services
App Inspection
Layout Inspector
Launch succeeded (4 minutes ago)
master


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
ReminderApp - single_reminder_file.xml [ReminderApp.app.main]
ReminderApp app src main res layout single_reminder_file.xml
Android
app
manifests
AndroidManifest.xml
java
com.dataflair.reminderapp
AlarmBrodcast
dbManager
MainActivity
Model
myAdapter
NotificationMessage
ReminderActivity
splashActivity
com.dataflair.reminderapp (androidTest)
com.dataflair.reminderapp (test)
java (generated)
com.dataflair.reminderapp
BuildConfig
res
drawable
layout
activity_main.xml
activity_notification_message.xml
activity_reminder.xml
activity_splash.xml
notification_layout.xml
single_reminder_file.xml
mipmap
values
res (generated)
Gradle Scripts
Model.java
activity_main.xml
activity_reminder.xml
activity_splash.xml
single_reminder_file.xml
Palette
Common
Text
Buttons
Widgets
Layouts
Containers
Helpers
Google
Legacy
TextView
Button
ImageView
RecyclerView
FragmentCo...
ScrollView
Switch
Component Tree
CardView
LinearLayout (vertical)
txtTitle "Title"
txtDate "Date"
txtTime "Time"
Attributes
Code
Split
Design
Emulator:
Pixel 4 API 30
1:35
Add Reminder Here...
I have to do Assignment
29-1-2023
1:35 AM
SAVE



1:36

I have to do Assignment

29-1-2023

1:35 AM

I'll Wake up

27-1-2023

3:37 AM

+

Successfully inserted