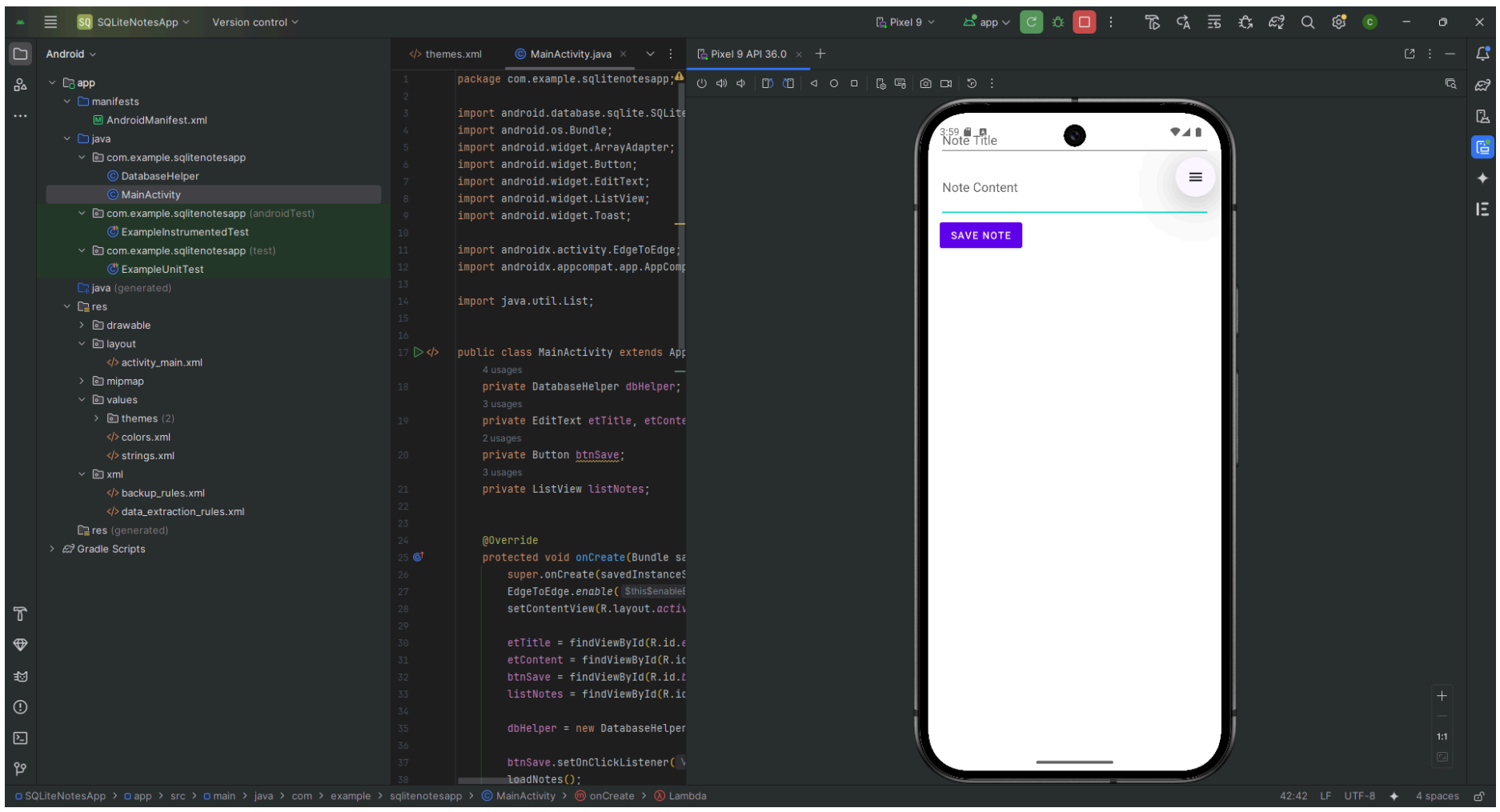
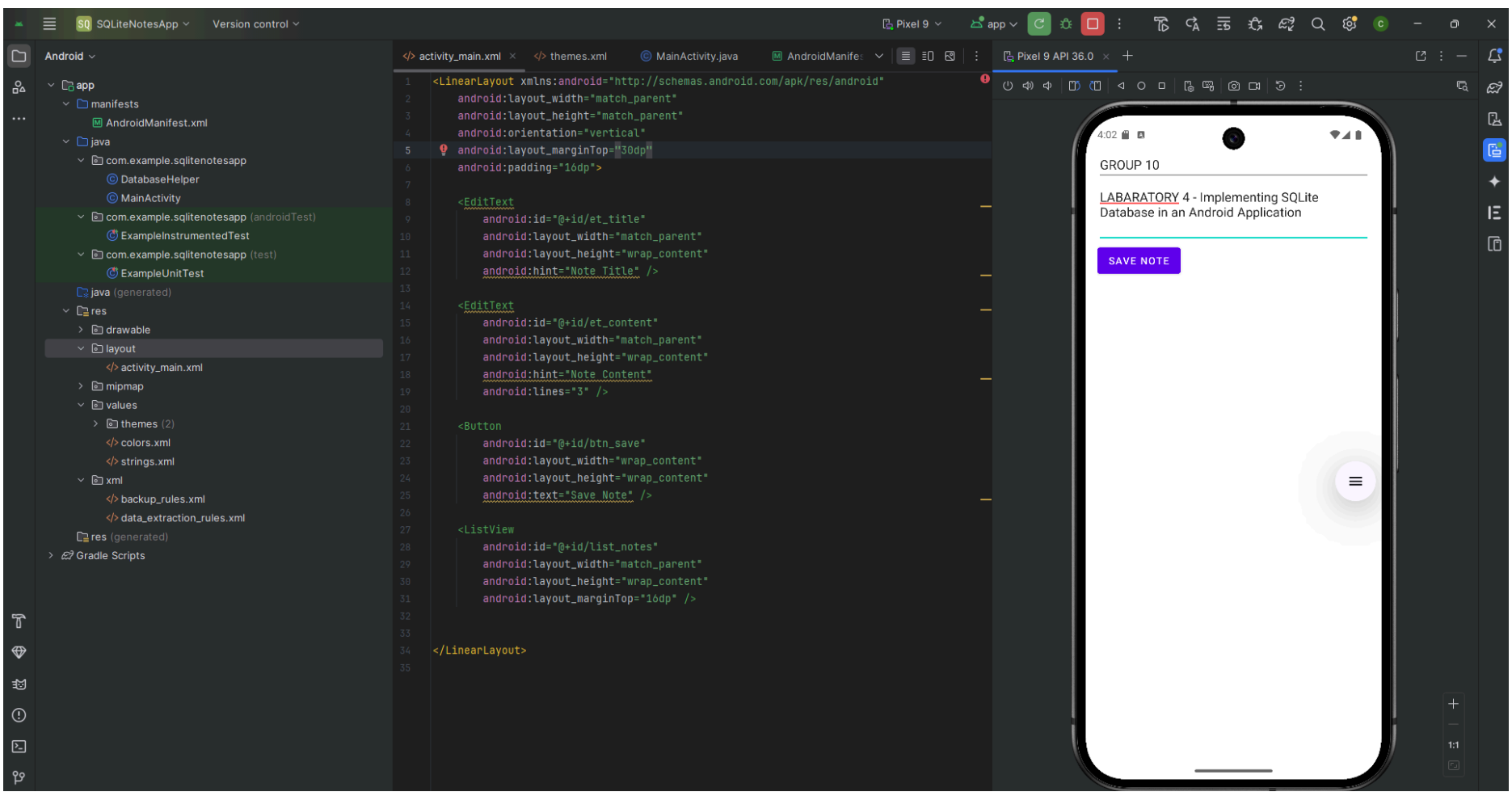
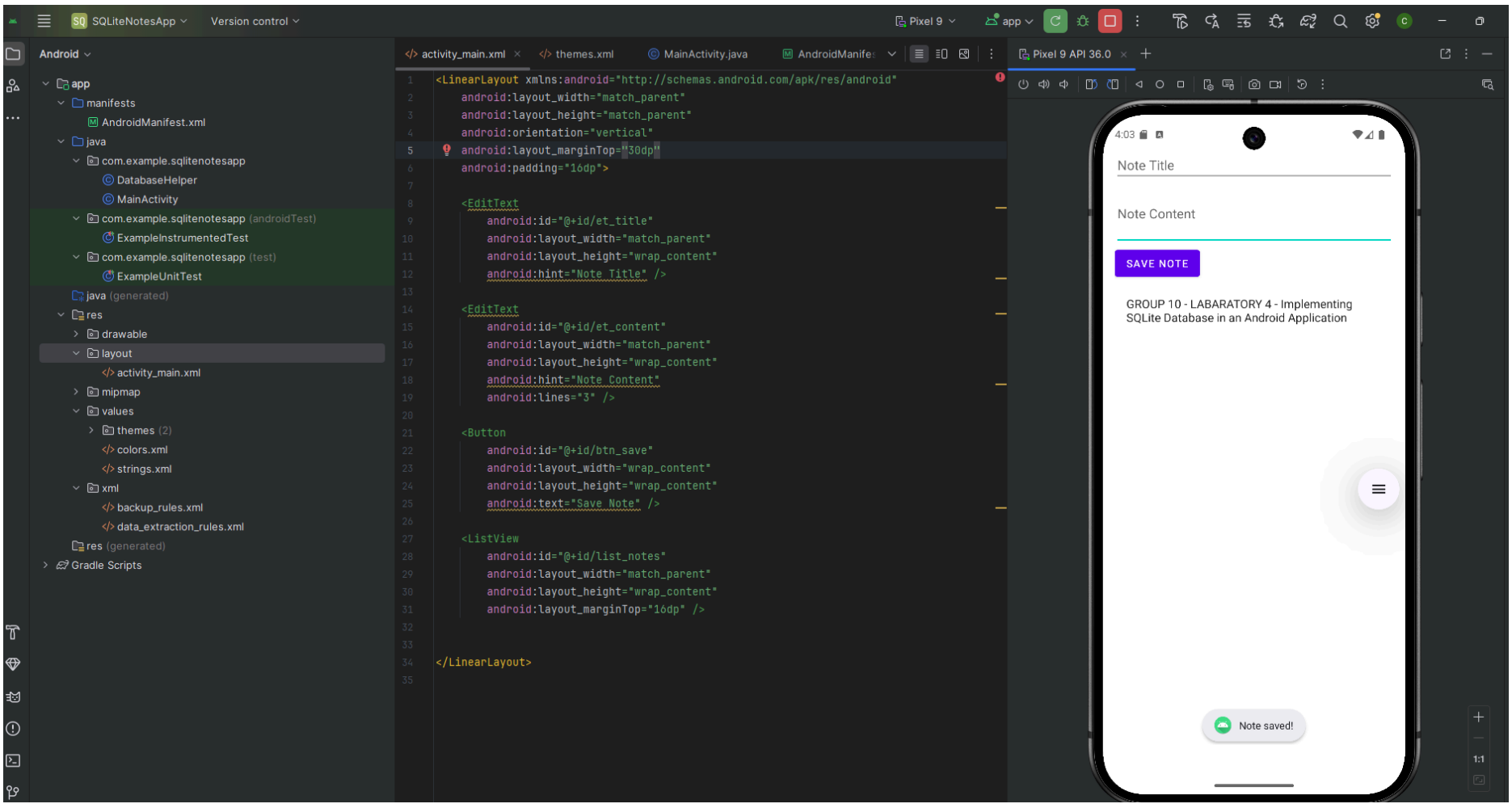
GROUP 10 / BSIT 3D Lb #4

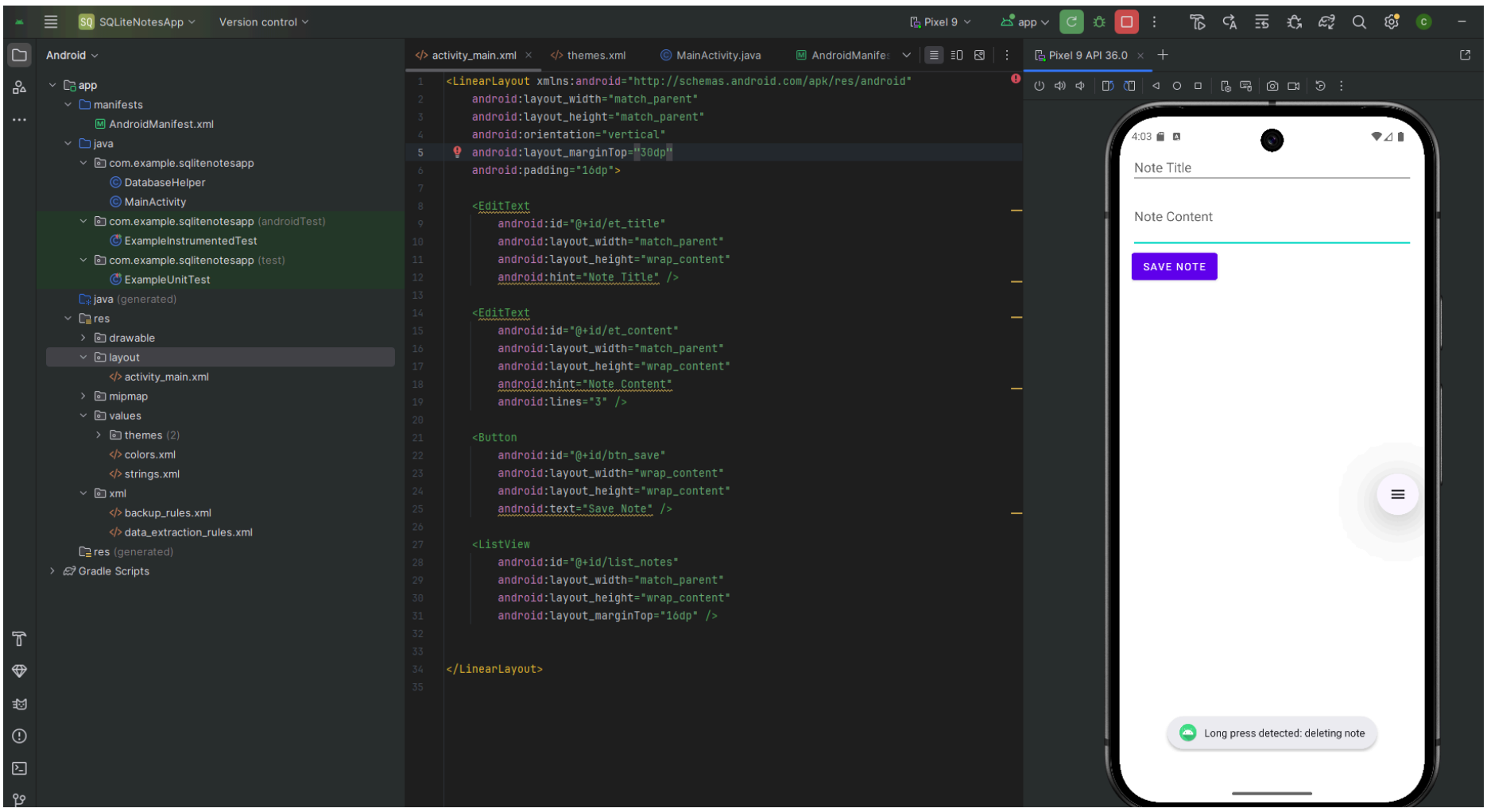
Implementing SQLite Database in an Android Application

Screenshots of your running app






SQLiteNotesApp
Version control
Pixel 9
app
Android
activity_main.xml
themes.xml
MainActivity.java
AndroidManifest.xml
Pixel 9 API 36.0
app
manifests
AndroidManifest.xml
java
com.example.sqlitenotesapp
DatabaseHelper
MainActivity
com.example.sqlitenotesapp (androidTest)
ExampleInstrumentedTest
com.example.sqlitenotesapp (test)
ExampleUnitTest
java (generated)
res
drawable
layout
activity_main.xml
mipmap
values
themes (2)
colors.xml
strings.xml
xml
backup_rules.xml
data_extraction_rules.xml
res (generated)
Gradle Scripts
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"
android:layout_marginTop="30dp"
android:padding="16dp">
<EditText
android:id="@+id/et_title"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Note Title" />
<EditText
android:id="@+id/et_content"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Note Content"
android:lines="3" />
<Button
android:id="@+id/btn_save"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Save Note" />
<ListView
android:id="@+id/list_notes"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginTop="16dp" />
</LinearLayout>
4:03
Note Title
Note Content
SAVE NOTE
GROUP 10 - LABARATORY 4 - Implementing SQLite Database in an Android Application
Note saved!
1:1


SQLiteNotesApp
Version control
Pixel 9
app
Android
activity_main.xml
themes.xml
MainActivity.java
AndroidManifest.xml
Pixel 9 API 36.0
app
manifests
AndroidManifest.xml
java
com.example.sqlitenotesapp
DatabaseHelper
MainActivity
com.example.sqlitenotesapp (androidTest)
ExampleInstrumentedTest
com.example.sqlitenotesapp (test)
ExampleUnitTest
java (generated)
res
drawable
layout
activity_main.xml
mipmap
values
themes (2)
colors.xml
strings.xml
xml
backup_rules.xml
data_extraction_rules.xml
res (generated)
Gradle Scripts
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"
android:layout_marginTop="30dp"
android:padding="16dp">
<EditText
android:id="@+id/et_title"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Note Title" />
<EditText
android:id="@+id/et_content"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Note Content"
android:lines="3" />
<Button
android:id="@+id/btn_save"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Save Note" />
<ListView
android:id="@+id/list_notes"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginTop="16dp" />
</LinearLayout>
4:03
Note Title
Note Content
SAVE NOTE
Long press detected: deleting note

Guide Questions:

1. **What is the purpose of the `SQLiteOpenHelper` class?**

   The SQLiteOpenHelper class helps manage database creation and upgrading, making it easier to open and maintain an SQLite database in an Android app.

2. **Explain the difference between `getReadableDatabase()` and `getWritableDatabase()`.**

   getReadableDatabase() provides a read-only database when writing isn't possible, while getWritableDatabase() provides full read-and-write access for inserting, updating, or deleting data.

3. **Why is it important to close the `Cursor` and `SQLiteDatabase` objects?**

   It is important to close the Cursor and SQLiteDatabase objects to prevent memory leaks, free system resources, and avoid database locks that could crash the app.

4. **How does the `ContentValues` class simplify data insertion?**

   The ContentValues class simplifies data insertion by allowing developers to store values as key-value pairs without writing raw SQL insert statements.

5. **What improvements can be added to make this app more user-friendly?**

   The app can be made more user-friendly by adding clear UI elements, error messages, search features, confirmation dialogs, and smoother feedback like Toast messages.