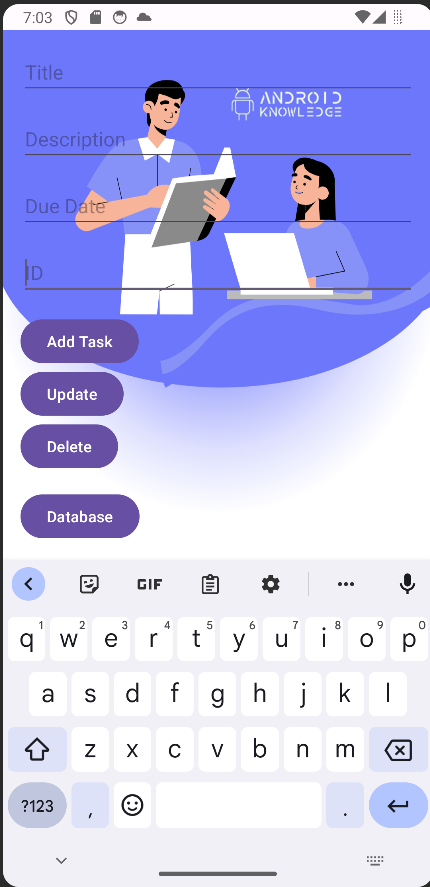
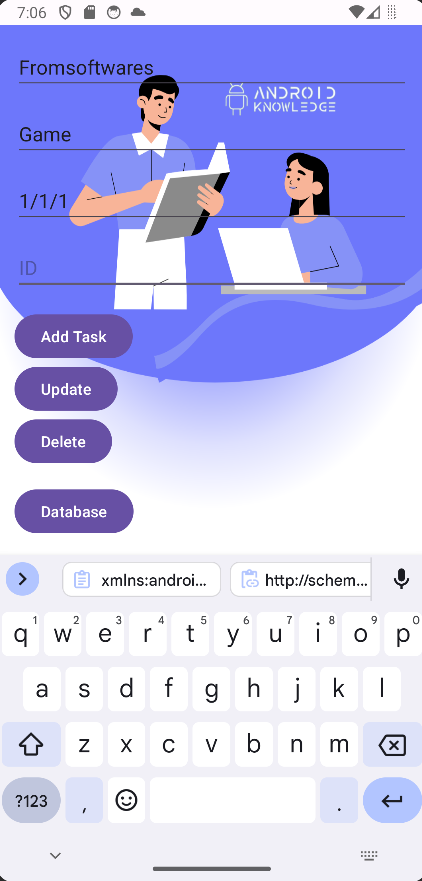
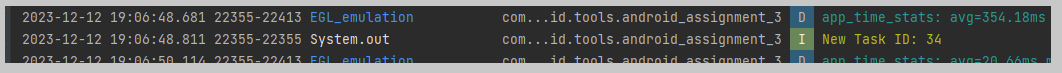
Screenshot:



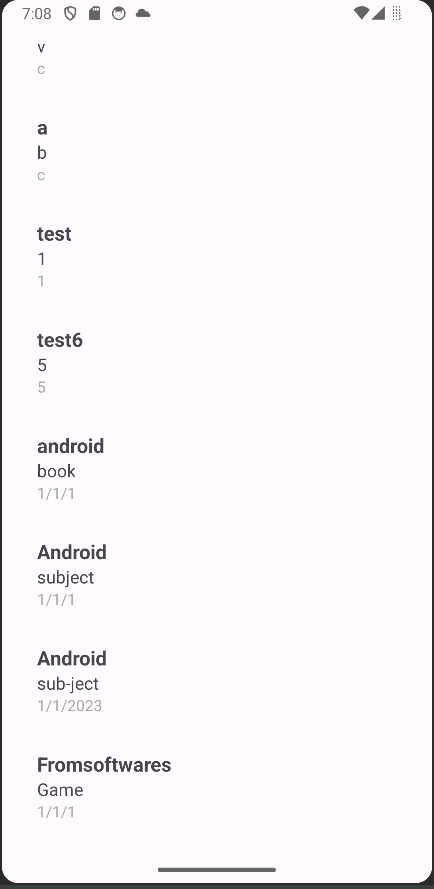
Adding Task:



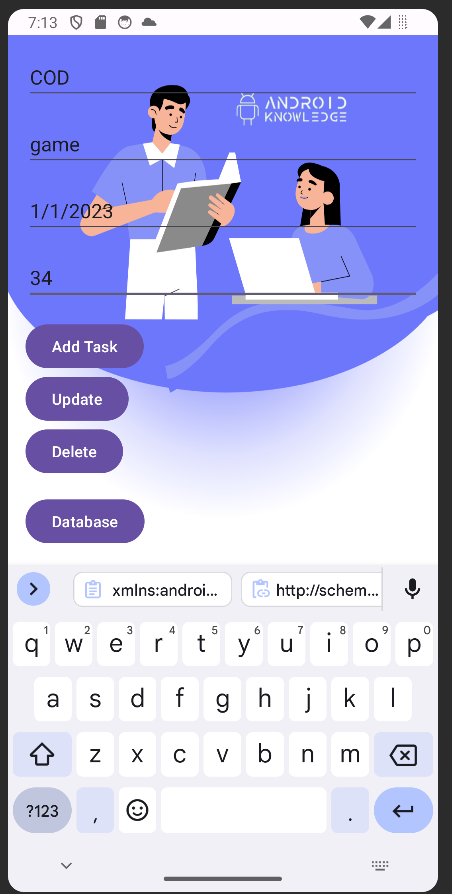
Saved With auto increamented ID: 34 which is current entries ID



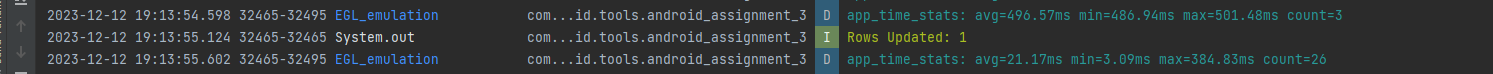
Showing in Database:



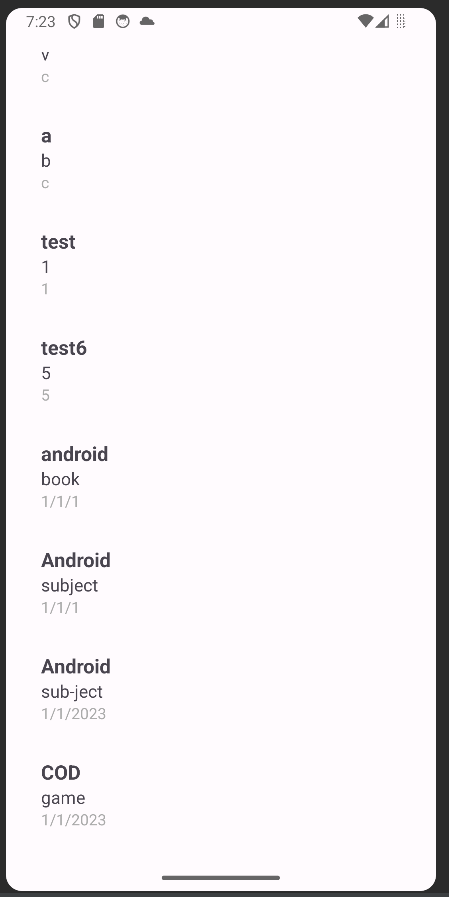
Updating now:



Logs:



Checking in Database:



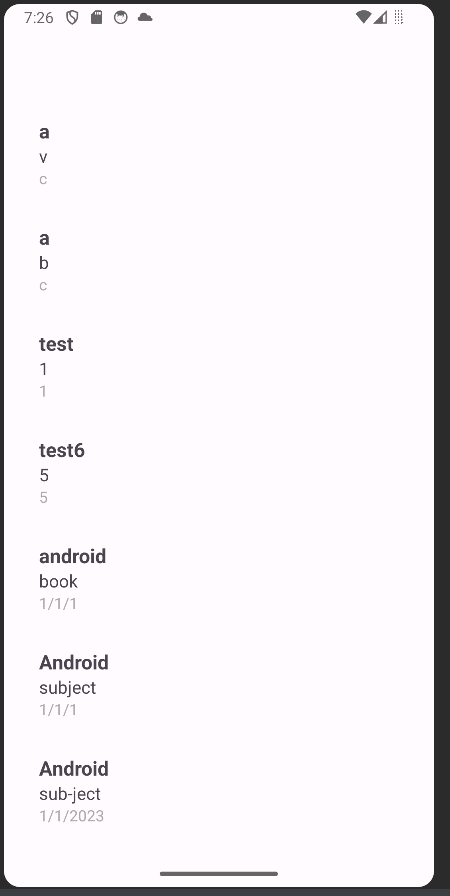
Deleting Now:



Logs:



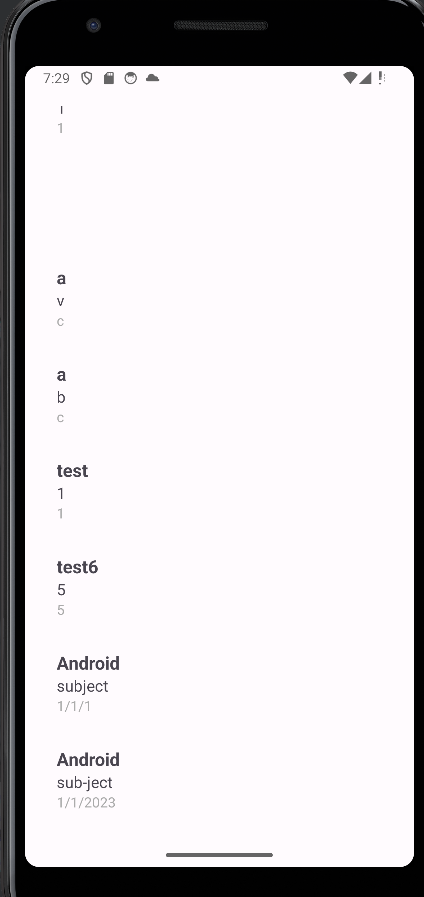
Database: As you can see the record has been deleted.



Deleting Task 30 at random:



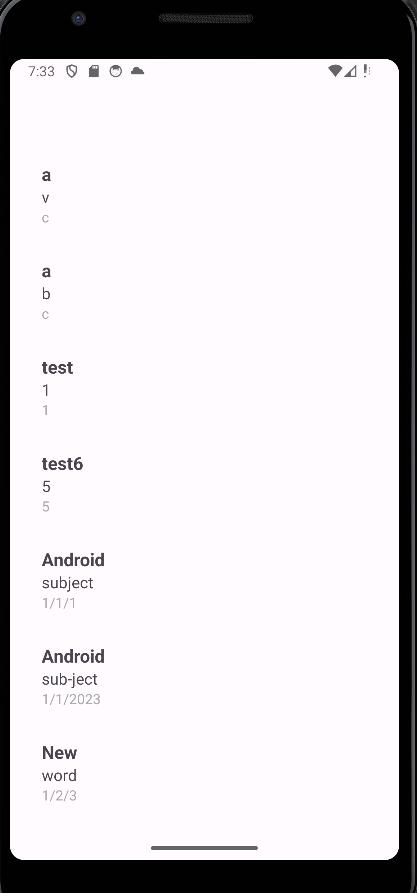




Adding another one:







Code:

MainActivity.java

| package com.android.tools.android\_assignment\_3;  import android.os.Bundle;  import android.content.Intent;  import android.view.View;  import android.widget.Button;  import android.widget.EditText;  import androidx.appcompat.app.AppCompatActivity;  public class MainActivity extends AppCompatActivity {  private TaskDatabaseHelper dbHelper;  private EditText editTitle;  private EditText editDescription;  private EditText editDueDate;  private EditText editTaskId;  private Button btnAdd;  private Button btnUpdate;  private Button btnDelete;  private Button btnDatabase;  private TaskListAdapter adapter;  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_main);  dbHelper = new TaskDatabaseHelper(this);  editTitle = findViewById(R.id.editTitle);  editDescription = findViewById(R.id.editDescription);  editDueDate = findViewById(R.id.editDueDate);  editTaskId = findViewById(R.id.editTaskId); // Assuming you have an edit text for task id  btnUpdate = findViewById(R.id.btnUpdate);  btnDelete = findViewById(R.id.btnDelete);  btnAdd = findViewById(R.id.btnAdd);  btnDatabase = findViewById(R.id.btnDatabase);  btnAdd.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  // Get values from EditText fields  String title = editTitle.getText().toString();  String description = editDescription.getText().toString();  String dueDate = editDueDate.getText().toString();  // Insert the task into the database  long newTaskId = dbHelper.addOrUpdateTask(title, description, dueDate, -1);  System.out.println("New Task ID: " + newTaskId);  }  });  btnDatabase.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  // Open TaskListActivity when "Database" button is clicked  startActivity(new Intent(MainActivity.this, TaskListActivity.class));  }  });  btnUpdate.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  // Get values from EditText fields  String title = editTitle.getText().toString();  String description = editDescription.getText().toString();  String dueDate = editDueDate.getText().toString();  // Get the task ID from user input  String idString = editTaskId.getText().toString();  if (idString.isEmpty()) {  // Handle the case where ID is empty  System.out.println("Task ID cannot be empty.");  return;  }  // Parse the user input to a long value  long taskIdToUpdate = Long.parseLong(idString);  // Create a new Task object with updated values  Task updatedTask = new Task(title, description, dueDate);  // Update the task in the database  long rowsUpdated = dbHelper.addOrUpdateTask(  updatedTask.getTitle(),  updatedTask.getDescription(),  updatedTask.getDueDate(),  taskIdToUpdate  );  System.out.println("Rows Updated: " + rowsUpdated);  }  });  btnDelete.setOnClickListener(new View.OnClickListener() {  @Override  public void onClick(View v) {  // Get the task ID from user input  String idString = editTaskId.getText().toString();  if (idString.isEmpty()) {  // Handle the case where ID is empty  System.out.println("Task ID cannot be empty.");  return;  }  // Parse the user input to a long value  long taskIdToDelete = Long.parseLong(idString);  // Delete the task from the database  dbHelper.deleteTask(taskIdToDelete);  System.out.println("Task deleted with ID: " + taskIdToDelete);  }  });  }  @Override  protected void onDestroy() {  // Close the database helper when the activity is destroyed  dbHelper.close();  super.onDestroy();  }  } |
| --- |

Task.java

| // Task.java  package com.android.tools.android\_assignment\_3;  public class Task {  private long id; // Assuming you have an ID field in your Task class  private String title;  private String description;  private String dueDate;  // Constructor with arguments  public Task(String title, String description, String dueDate) {  this.title = title;  this.description = description;  this.dueDate = dueDate;  }  public Task() {  // Provide default values or leave fields uninitialized  }  // Getters and setters for other fields, if needed  public long getId() {  return id;  }  public void setId(long id) {  this.id = id;  }  public String getTitle() {  return title;  }  public void setTitle(String title) {  this.title = title;  }  public String getDescription() {  return description;  }  public void setDescription(String description) {  this.description = description;  }  public String getDueDate() {  return dueDate;  }  public void setDueDate(String dueDate) {  this.dueDate = dueDate;  }  } |
| --- |

TaskDatabaseHelper:

| package com.android.tools.android\_assignment\_3;  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;  import java.util.List;  public class TaskDatabaseHelper extends SQLiteOpenHelper {  private static final String DATABASE\_NAME = "tasks.db";  private static final int DATABASE\_VERSION = 1;  // Renamed to avoid duplication  private static final String TASKS\_TABLE = "tasks";  // Renamed to avoid duplication  public static final String COLUMN\_ID = "\_id";  public static final String COLUMN\_TITLE = "title";  public static final String COLUMN\_DESCRIPTION = "description";  public static final String COLUMN\_DUE\_DATE = "due\_date";  // Database creation SQL statement  private static final String DATABASE\_CREATE = "create table "  + TASKS\_TABLE + "(" + COLUMN\_ID  + " integer primary key autoincrement, " + COLUMN\_TITLE  + " text not null, " + COLUMN\_DESCRIPTION  + " text not null, " + COLUMN\_DUE\_DATE + " text not null);";  public TaskDatabaseHelper(Context context) {  super(context, DATABASE\_NAME, null, DATABASE\_VERSION);  }  @Override  public void onCreate(SQLiteDatabase database) {  database.execSQL(DATABASE\_CREATE);  }  @Override  public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  db.execSQL("DROP TABLE IF EXISTS " + TASKS\_TABLE);  onCreate(db);  }  public long addOrUpdateTask(String title, String description, String dueDate, long taskId) {  SQLiteDatabase db = this.getWritableDatabase();  ContentValues values = new ContentValues();  values.put(COLUMN\_TITLE, title);  values.put(COLUMN\_DESCRIPTION, description);  values.put(COLUMN\_DUE\_DATE, dueDate);  if (taskId == -1) {  return db.insert(TASKS\_TABLE, null, values);  } else {  return db.update(TASKS\_TABLE, values, COLUMN\_ID + " = ?",  new String[]{String.valueOf(taskId)});  }  }  public void deleteTask(long taskId) {  SQLiteDatabase db = this.getWritableDatabase();  db.delete(TASKS\_TABLE, COLUMN\_ID + " = ?",  new String[]{String.valueOf(taskId)});  db.close();  }  public Task getTaskByDescription(String description) {  SQLiteDatabase db = this.getReadableDatabase();  Task task = null;  Cursor cursor = db.query(  TASKS\_TABLE,  new String[]{COLUMN\_ID, COLUMN\_TITLE, COLUMN\_DESCRIPTION, COLUMN\_DUE\_DATE},  COLUMN\_DESCRIPTION + " = ?",  new String[]{description},  null, null, null  );  if (cursor != null && cursor.moveToFirst()) {  task = new Task();  task.setId(cursor.getLong(cursor.getColumnIndex(COLUMN\_ID)));  task.setTitle(cursor.getString(cursor.getColumnIndex(COLUMN\_TITLE)));  task.setDescription(cursor.getString(cursor.getColumnIndex(COLUMN\_DESCRIPTION)));  task.setDueDate(cursor.getString(cursor.getColumnIndex(COLUMN\_DUE\_DATE)));  cursor.close();  }  return task;  }  public List<Task> getAllTasks() {  List<Task> taskList = new ArrayList<>();  SQLiteDatabase db = this.getReadableDatabase();  Cursor cursor = db.query(TASKS\_TABLE, null, null, null, null, null, null);  if (cursor != null && cursor.moveToFirst()) {  do {  Task task = new Task();  task.setId(cursor.getLong(cursor.getColumnIndex(COLUMN\_ID)));  task.setTitle(cursor.getString(cursor.getColumnIndex(COLUMN\_TITLE)));  task.setDescription(cursor.getString(cursor.getColumnIndex(COLUMN\_DESCRIPTION)));  task.setDueDate(cursor.getString(cursor.getColumnIndex(COLUMN\_DUE\_DATE)));  taskList.add(task);  } while (cursor.moveToNext());  cursor.close();  }  return taskList;  }  } |
| --- |

TaskListActivity:

| package com.android.tools.android\_assignment\_3;  import android.os.Bundle;  import androidx.appcompat.app.AppCompatActivity;  import androidx.recyclerview.widget.LinearLayoutManager;  import androidx.recyclerview.widget.RecyclerView;  import java.util.List;  public class TaskListActivity extends AppCompatActivity {  private TaskDatabaseHelper dbHelper;  private RecyclerView recyclerView;  private TaskListAdapter adapter; // Change the type to TaskListAdapter  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  setContentView(R.layout.activity\_task\_list);  dbHelper = new TaskDatabaseHelper(this);  recyclerView = findViewById(R.id.recyclerView);  adapter = new TaskListAdapter(); // Initialize the adapter with TaskListAdapter  // Set up RecyclerView  recyclerView.setLayoutManager(new LinearLayoutManager(this));  recyclerView.setAdapter(adapter);  // Load tasks from the database  loadTasks();  }  private void loadTasks() {  List<Task> taskList = dbHelper.getAllTasks();  adapter.setTasks(taskList);  }  } |
| --- |

TaskListAdapter.java

| // TaskListAdapter.java  package com.android.tools.android\_assignment\_3;  import android.view.LayoutInflater;  import android.view.View;  import android.view.ViewGroup;  import android.widget.TextView;  import androidx.annotation.NonNull;  import androidx.recyclerview.widget.RecyclerView;  import java.util.List;  public class TaskListAdapter extends RecyclerView.Adapter<TaskListAdapter.TaskViewHolder> {  private List<Task> taskList;  public void setTasks(List<Task> taskList) {  this.taskList = taskList;  notifyDataSetChanged();  }  @NonNull  @Override  public TaskViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  View view = LayoutInflater.from(parent.getContext()).inflate(R.layout.task\_list\_item, parent, false);  return new TaskViewHolder(view);  }  @Override  public void onBindViewHolder(@NonNull TaskViewHolder holder, int position) {  if (taskList != null) {  Task task = taskList.get(position);  holder.bind(task);  }  }  @Override  public int getItemCount() {  return taskList != null ? taskList.size() : 0;  }  static class TaskViewHolder extends RecyclerView.ViewHolder {  private TextView titleTextView;  private TextView descriptionTextView;  private TextView dueDateTextView;  public TaskViewHolder(@NonNull View itemView) {  super(itemView);  titleTextView = itemView.findViewById(R.id.titleTextView);  descriptionTextView = itemView.findViewById(R.id.descriptionTextView);  dueDateTextView = itemView.findViewById(R.id.dueDateTextView);  }  public void bind(Task task) {  titleTextView.setText(task.getTitle());  descriptionTextView.setText(task.getDescription());  dueDateTextView.setText(task.getDueDate());  }  }  } |
| --- |

activitymain.xml

| <!-- activity\_main.xml -->  <LinearLayout  xmlns:android="http://schemas.android.com/apk/res/android"  xmlns:tools="http://schemas.android.com/tools"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:orientation="vertical"  android:padding="@dimen/activity\_horizontal\_margin"  android:background="@drawable/img"  tools:context=".MainActivity">  <EditText  android:id="@+id/editTitle"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:hint="Title" />  <EditText  android:id="@+id/editDescription"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="16dp"  android:hint="Description" />  <EditText  android:id="@+id/editDueDate"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="16dp"  android:hint="Due Date" />  <EditText  android:id="@+id/editTaskId"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:layout\_marginTop="16dp"  android:hint="ID" />  <Button  android:id="@+id/btnAdd"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginTop="16dp"  android:text="Add Task" />  <Button  android:id="@+id/btnUpdate"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Update" />  <Button  android:id="@+id/btnDelete"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Delete" />  <Button  android:id="@+id/btnDatabase"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_marginTop="16dp"  android:text="Database" />  </LinearLayout> |
| --- |

task\_list\_item.xml

| <?xml version="1.0" encoding="utf-8"?>  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:orientation="vertical"  android:padding="16dp">  <TextView  android:id="@+id/titleTextView"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="18sp"  android:textStyle="bold" />  <TextView  android:id="@+id/descriptionTextView"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="16sp" />  <TextView  android:id="@+id/dueDateTextView"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:textSize="14sp"  android:textColor="@android:color/darker\_gray" />  </LinearLayout> |
| --- |