**1. Create an application of todo app using SQLite with function lite to create list of upcoming task, completed task, remove task, update task in daily activity.**

**Step:1 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="16dp"

tools:context=".MainActivity">

<EditText

android:id="@+id/editTextTask"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Enter a task" />

<Button

android:id="@+id/buttonAddTask"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Add Task"

android:layout\_marginTop="8dp"

android:onClick="addTask" />

<TextView

android:id="@+id/textViewUpcomingTasks"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Upcoming Tasks"

android:textStyle="bold"

android:layout\_marginTop="16dp" />

<ListView

android:id="@+id/listViewUpcomingTasks"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp" />

<TextView

android:id="@+id/textViewCompletedTasks"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Completed Tasks"

android:textStyle="bold"

android:layout\_marginTop="16dp" />

<ListView

android:id="@+id/listViewCompletedTasks"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="8dp" />

</LinearLayout>

**Step:2 list\_item.xml**

<TextView

xmlns:android="http://schemas.android.com/apk/res/android"

android:id="@+id/textViewTaskItem"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="8dp" />

**Step:3 dataclass**

data class Task(val id: Long, var name: String, var isCompleted: Boolean)

**Step:4 MainActivity.kt**

import android.os.Bundle

import android.view.View

import android.widget.\*

class MainActivity : AppCompatActivity() {

private lateinit var db: SQLiteDatabase

private lateinit var upcomingTasksListView: ListView

private lateinit var completedTasksListView: ListView

private lateinit var taskAdapter: ArrayAdapter<String>

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

val taskDbHelper = TaskDbHelper(this)

db = taskDbHelper.writableDatabase

upcomingTasksListView = findViewById(R.id.listViewUpcomingTasks)

completedTasksListView = findViewById(R.id.listViewCompletedTasks)

refreshTaskLists()

upcomingTasksListView.onItemClickListener = AdapterView.OnItemClickListener { \_, \_, position, \_ ->

val taskId = upcomingTasksListView.getItemAtPosition(position) as Long

updateTaskCompletion(taskId, true)

refreshTaskLists()

}

completedTasksListView.onItemClickListener = AdapterView.OnItemClickListener { \_, \_, position, \_ ->

val taskId = completedTasksListView.getItemAtPosition(position) as Long

updateTaskCompletion(taskId, false)

refreshTaskLists()

}

}

fun addTask(view: View) {

val taskEditText = findViewById<EditText>(R.id.editTextTask)

val taskName = taskEditText.text.toString().trim()

if (taskName.isNotEmpty()) {

insertTask(taskName)

taskEditText.text.clear()

refreshTaskLists()

}

}

private fun insertTask(name: String) {

val values = ContentValues().apply {

put(TaskContract.TaskEntry.COLUMN\_NAME, name)

put(TaskContract.TaskEntry.COLUMN\_COMPLETED, false)

}

db.insert(TaskContract.TaskEntry.TABLE\_NAME, null, values)

}

private fun updateTaskCompletion(id: Long, isCompleted: Boolean) {

val values = ContentValues().apply {

put(TaskContract.TaskEntry.COLUMN\_COMPLETED, isCompleted)

}

val selection = "${TaskContract.TaskEntry.\_ID} = ?"

val selectionArgs = arrayOf(id.toString())

db.update(

TaskContract.TaskEntry.TABLE\_NAME,

values,

selection,

selectionArgs

)

}

private fun refreshTaskLists() {

val upcomingTasks = mutableListOf<Long>()

val completedTasks = mutableListOf<Long>()

val projection = arrayOf(

TaskContract.TaskEntry.\_ID,

TaskContract.TaskEntry.COLUMN\_NAME,

TaskContract.TaskEntry.COLUMN\_COMPLETED

)

val sortOrder = "${TaskContract.TaskEntry.\_ID} ASC"

val cursor = db.query(

TaskContract.TaskEntry.TABLE\_NAME,

projection,

null,

null,

null,

null,

sortOrder

)

while (cursor.moveToNext()) {

val id = cursor.getLong(cursor.getColumnIndexOrThrow(TaskContract.TaskEntry.\_ID))

val name = cursor.getString(cursor.getColumnIndexOrThrow(TaskContract.TaskEntry.COLUMN\_NAME))

val isCompleted = cursor.getInt(cursor.getColumnIndexOrThrow(TaskContract.TaskEntry.COLUMN\_COMPLETED)) == 1

if (isCompleted) {

completedTasks.add(id)

} else {

upcomingTasks.add(id)

}

}

cursor.close()

val upcomingTasksAdapter = createTaskAdapter(upcomingTasks)

upcomingTasksListView.adapter = upcomingTasksAdapter

val completedTasksAdapter = createTaskAdapter(completedTasks)

completedTasksListView.adapter = completedTasksAdapter

}

private fun createTaskAdapter(tasks: List<Long>): ArrayAdapter<String> {

val taskNames = mutableListOf<String>()

for (taskId in tasks) {

val taskCursor = db.query(

TaskContract.TaskEntry.TABLE\_NAME,

arrayOf(TaskContract.TaskEntry.COLUMN\_NAME),

"${TaskContract.TaskEntry.\_ID} = ?",

arrayOf(taskId.toString()),

null,

null,

null

)

if (taskCursor.moveToFirst()) {

val taskName = taskCursor.getString(taskCursor.getColumnIndexOrThrow(TaskContract.TaskEntry.COLUMN\_NAME))

taskNames.add(taskName)

}

taskCursor.close()

}

taskAdapter = ArrayAdapter(this, R.layout.list\_item\_task, taskNames)

return taskAdapter

}

override fun onDestroy() {

super.onDestroy()

db.close()

}

}

**Step:4 TaskContract.kt**

import android.provider.BaseColumns

object TaskContract {

object TaskEntry : BaseColumns {

const val TABLE\_NAME = "tasks"

const val COLUMN\_NAME = "name"

const val COLUMN\_COMPLETED = "completed"

}

}

Step:5 TaskDbHelper.kt

import android.content.Context

import android.database.sqlite.SQLiteDatabase

import android.database.sqlite.SQLiteOpenHelper

class TaskDbHelper(context: Context) : SQLiteOpenHelper(context, DATABASE\_NAME, null, DATABASE\_VERSION) {

override fun onCreate(db: SQLiteDatabase) {

val createTableQuery = """

CREATE TABLE ${TaskContract.TaskEntry.TABLE\_NAME} (

${TaskContract.TaskEntry.\_ID} INTEGER PRIMARY KEY AUTOINCREMENT,

${TaskContract.TaskEntry.COLUMN\_NAME} TEXT NOT NULL,

${TaskContract.TaskEntry.COLUMN\_COMPLETED} INTEGER NOT NULL

)

""".trimIndent()

db.execSQL(createTableQuery)

}

override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {

val deleteTableQuery = "DROP TABLE IF EXISTS ${TaskContract.TaskEntry.TABLE\_NAME}"

db.execSQL(deleteTableQuery)

onCreate(db)

}

companion object {

private const val DATABASE\_NAME = "task.db"

private const val DATABASE\_VERSION = 1

}

}

**2. Create a Sticky Notes app with proper customization which can insert,view,update,delete using SQLite database.**

**Step1: activity\_main.xml**

<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">

<EditText

android:id="@+id/editTextTitle"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Title"

android:textSize="18sp" />

<EditText

android:id="@+id/editTextContent"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Content"

android:textSize="16sp" />

<Button

android:id="@+id/buttonSave"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Save" />

<Button

android:id="@+id/buttonDelete"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Delete" />

</LinearLayout>

**Step2: data class**

data class Note(val id: Int, var title: String, var content: String)

**Step3: NoteAdapter.kt**

import android.view.LayoutInflater

import android.view.View

import android.view.ViewGroup

import androidx.recyclerview.widget.RecyclerView

import kotlinx.android.synthetic.main.note\_item.view.\*

class NoteAdapter(private val notes: List<Note>) :

RecyclerView.Adapter<NoteAdapter.NoteViewHolder>() {

class NoteViewHolder(itemView: View) : RecyclerView.ViewHolder(itemView)

override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): NoteViewHolder {

val itemView = LayoutInflater.from(parent.context)

.inflate(R.layout.note\_item, parent, false)

return NoteViewHolder(itemView)

}

override fun onBindViewHolder(holder: NoteViewHolder, position: Int) {

val currentNote = notes[position]

holder.itemView.editTextTitle.setText(currentNote.title)

holder.itemView.editTextContent.setText(currentNote.content)

}

override fun getItemCount() = notes.size

}

**Step4: ActivityMain.kt**

import android.content.ContentValues

import android.database.Cursor

import android.database.sqlite.SQLiteDatabase

import android.os.Bundle

import android.view.LayoutInflater

import android.view.View

import android.view.ViewGroup

import androidx.appcompat.app.AppCompatActivity

import androidx.recyclerview.widget.LinearLayoutManager

import kotlinx.android.synthetic.main.activity\_main.\*

import kotlinx.android.synthetic.main.note\_item.view.\*

class MainActivity : AppCompatActivity() {

private lateinit var db: SQLiteDatabase

private val notes: MutableList<Note> = mutableListOf()

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

val dbHelper = NoteDbHelper(this)

db = dbHelper.writableDatabase

recyclerViewNotes.layoutManager = LinearLayoutManager(this)

recyclerViewNotes.adapter = NoteAdapter(notes)

buttonSave.setOnClickListener {

val title = editTextTitle.text.toString().trim()

val content = editTextContent.text.toString().trim()

if (title.isNotEmpty() && content.isNotEmpty()) {

val note = Note(

id = -1,

title = title,

content = content

)

insertNoteIntoDb(note)

clearFields()

loadNotesFromDb()

}

}

buttonDelete.setOnClickListener {

val selectedNote = notes.getOrNull(recyclerViewNotes.selectedItemId.toInt())

selectedNote?.let { deleteNoteFromDb(it) }

loadNotesFromDb()

}

loadNotesFromDb()

}

override fun onDestroy() {

super.onDestroy()

db.close()

}

private fun insertNoteIntoDb(note: Note) {

val values = ContentValues().apply {

put(NoteContract.NoteEntry.COLUMN\_TITLE, note.title)

put(NoteContract.NoteEntry.COLUMN\_CONTENT, note.content)

}

db.insert(NoteContract.NoteEntry.TABLE\_NAME, null, values)

}

private fun loadNotesFromDb() {

notes.clear()

val projection = arrayOf(

NoteContract.NoteEntry.\_ID,

NoteContract.NoteEntry.COLUMN\_TITLE,

NoteContract.NoteEntry.COLUMN\_CONTENT

)

val sortOrder = "${NoteContract.NoteEntry.\_ID} DESC"

val cursor: Cursor = db.query(

NoteContract.NoteEntry.TABLE\_NAME,

projection,

null,

null,

null,

null,

sortOrder

)

with(cursor) {

while (moveToNext()) {

val id = getInt(getColumnIndexOrThrow(NoteContract.NoteEntry.\_ID))

val title =

getString(getColumnIndexOrThrow(NoteContract.NoteEntry.COLUMN\_TITLE))

val content =

getString(getColumnIndexOrThrow(NoteContract.NoteEntry.COLUMN\_CONTENT))

notes.add(Note(id, title, content))

}

}

recyclerViewNotes.adapter?.notifyDataSetChanged()

}

private fun deleteNoteFromDb(note: Note) {

val selection = "${NoteContract.NoteEntry.\_ID} LIKE ?"

val selectionArgs = arrayOf(note.id.toString())

db.delete(NoteContract.NoteEntry.TABLE\_NAME, selection, selectionArgs)

}

private fun clearFields() {

editTextTitle.text.clear()

editTextContent.text.clear()

}

}

Step5: NoteDbHelper (class file)

import android.content.Context

import android.database.sqlite.SQLiteDatabase

import android.database.sqlite.SQLiteOpenHelper

class NoteDbHelper(context: Context) :

SQLiteOpenHelper(context, DATABASE\_NAME, null, DATABASE\_VERSION) {

override fun onCreate(db: SQLiteDatabase) {

val createTableQuery = """

CREATE TABLE ${NoteContract.NoteEntry.TABLE\_NAME} (

${NoteContract.NoteEntry.\_ID} INTEGER PRIMARY KEY AUTOINCREMENT,

${NoteContract.NoteEntry.COLUMN\_TITLE} TEXT NOT NULL,

${NoteContract.NoteEntry.COLUMN\_CONTENT} TEXT NOT NULL

)

""".trimIndent()

db.execSQL(createTableQuery)

}

override fun onUpgrade(db: SQLiteDatabase, oldVersion: Int, newVersion: Int) {

val deleteTableQuery =

"DROP TABLE IF EXISTS ${NoteContract.NoteEntry.TABLE\_NAME}"

db.execSQL(deleteTableQuery)

onCreate(db)

}

companion object {

private const val DATABASE\_NAME = "notes.db"

private const val DATABASE\_VERSION = 1

}

}

Step6: NoteContract (class)

object NoteContract {

object NoteEntry {

const val TABLE\_NAME = "notes"

const val \_ID = "\_id"

const val COLUMN\_TITLE = "title"

const val COLUMN\_CONTENT = "content"

}

}

**3. create task management application for adding, updating, deleting the task and show the tasks in the listview or gridview. Task have name, description, date, time, priority. Sort the task by the date and time wise. If task is due then automatically show as blue color. It will search the tasks by date wise. If high priority then show as red color, average priority as blue color, low priority as green color. Select specified item an open context menu to select “Complete the Task” then this task.**

**Step1: activity\_main.xml**

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout 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"

tools:context=".MainActivity">

<ListView

android:id="@+id/listViewTasks"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" />

</RelativeLayout>

Step2: list\_item\_task.xml

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:padding="8dp">

<TextView

android:id="@+id/textViewName"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textSize="16sp" />

<TextView

android:id="@+id/textViewDescription"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@id/textViewName"

android:textSize="14sp" />

<TextView

android:id="@+id/textViewDateTime"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_below="@id/textViewDescription"

android:textSize="12sp" />

</RelativeLayout>

Step3: Task.kt (data class)

data class Task(

val name: String,

val description: String,

val date: String,

val time: String,

val priority: String

)

Step4: MainAvtivity.kt

import android.graphics.Color

import android.os.Bundle

import android.view.ContextMenu

import android.view.MenuItem

import android.view.View

import android.widget.AdapterView

import android.widget.ArrayAdapter

import android.widget.ListView

import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {

private lateinit var listViewTasks: ListView

private lateinit var tasks: MutableList<Task>

private lateinit var adapter: ArrayAdapter<Task>

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_main)

listViewTasks = findViewById(R.id.listViewTasks)

tasks = mutableListOf()

adapter = ArrayAdapter(this, R.layout.list\_item\_task, tasks)

listViewTasks.adapter = adapter

listViewTasks.setOnItemClickListener { \_, \_, position, \_ ->

val task = tasks[position]

// Handle task item click event

}

registerForContextMenu(listViewTasks)

loadTasks()

}

override fun onCreateContextMenu(

menu: ContextMenu,

v: View,

menuInfo: ContextMenu.ContextMenuInfo?

) {

super.onCreateContextMenu(menu, v, menuInfo)

menuInflater.inflate(R.menu.context\_menu\_task, menu)

}

override fun onContextItemSelected(item: MenuItem): Boolean {

val info = item.menuInfo as AdapterView.AdapterContextMenuInfo

val position = info.position

val task = tasks[position]

return when (item.itemId) {

R.id.menuComplete -> {

// Handle complete task action

true

}

else -> super.onContextItemSelected(item)

}

}

private fun loadTasks() {

// Load tasks from the database or any other data source

// You can replace this with your implementation using SQLite

tasks.add(Task("Task 1", "Description 1", "2023-06-07", "10:00 AM", "High"))

tasks.add(Task("Task 2", "Description 2", "2023-06-08", "2:00 PM", "Medium"))

tasks.add(Task("Task 3", "Description 3", "2023-06-09", "9:00 AM", "Low"))

sortTasksByDateTime()

adapter.notifyDataSetChanged()

}

private fun sortTasksByDateTime() {

tasks.sortBy { it.date + " " + it.time }

}

private fun colorCodeTaskItem(task: Task) {

val position = tasks.indexOf(task)

val view = listViewTasks.getChildAt(position)

view?.apply {

val priorityColor = when (task.priority.toLowerCase()) {

"high" -> Color.RED

"medium" -> Color.BLUE

"low" -> Color.GREEN

else -> Color.BLACK

}

setBackgroundColor(priorityColor)

}

}

}

Step5: context\_menu\_task.xml

<menu xmlns:android="http://schemas.android.com/apk/res/android">

<item

android:id="@+id/menuComplete"

android:title="Complete the Task" />

</menu>

**4. Create an application in which employee can login and register with MySQL database.**

**Step1: Set up MySQL Database:**

Create a database : “employee management”

Create table “employee”{“id” (INT, AUTO\_INCREMENT),”username”(VARCHAR), “password”(VARCHAR)}

**Step2: activity\_main.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="match\_parent"

android:orientation="vertical"

android:padding="16dp">

<EditText

android:id="@+id/editTextUsername"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Username" />

<EditText

android:id="@+id/editTextPassword"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="Password"

android:inputType="textPassword" />

<Button

android:id="@+id/buttonLogin"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Login" />

<Button

android:id="@+id/buttonRegister"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Register" />

</LinearLayout>

**Step3: activity\_register.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="match\_parent"

android:orientation="vertical"

android:padding="16dp">

<EditText

android:id="@+id/editTextNewUsername"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="New Username" />

<EditText

android:id="@+id/editTextNewPassword"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:hint="New Password"

android:inputType="textPassword" />

<Button

android:id="@+id/buttonRegisterNew"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Register" />

</LinearLayout>

**Step4: Employee.kt (data class)**

data class Employee(

val id: Int,

val username: String,

val password: String

)

**Step5: MainAvtivity.kt**

import android.app.Activity

import android.content.Intent

import android.graphics.Bitmap

import android.os.Bundle

import android.provider.MediaStore

import android.view.View

import android.widget.Button

import android.widget.ImageView

import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {

private lateinit var imageViewProfile: ImageView

private lateinit var buttonUpload: Button

companion object {

private const val REQUEST\_IMAGE\_CAPTURE = 1

}

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_profile)

imageViewProfile = findViewById(R.id.imageViewProfile)

buttonUpload = findViewById(R.id.buttonUpload)

buttonUpload.setOnClickListener {

dispatchTakePictureIntent()

}

}

private fun dispatchTakePictureIntent() {

val takePictureIntent = Intent(MediaStore.ACTION\_IMAGE\_CAPTURE)

if (takePictureIntent.resolveActivity(packageManager) != null) {

startActivityForResult(takePictureIntent, REQUEST\_IMAGE\_CAPTURE)

}

}

override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {

super.onActivityResult(requestCode, resultCode, data)

if (requestCode == REQUEST\_IMAGE\_CAPTURE && resultCode == Activity.RESULT\_OK) {

val imageBitmap = data?.extras?.get("data") as Bitmap

imageViewProfile.setImageBitmap(imageBitmap)

}

}

}

**Step6: RegisterActivity.kt**

import android.os.Bundle

import android.view.View

import android.widget.Button

import android.widget.EditText

import androidx.appcompat.app.AppCompatActivity

class RegisterActivity : AppCompatActivity() {

private lateinit var editTextNewUsername: EditText

private lateinit var editTextNewPassword: EditText

private lateinit var buttonRegisterNew: Button

override fun onCreate(savedInstanceState: Bundle?) {

super.onCreate(savedInstanceState)

setContentView(R.layout.activity\_register)

editTextNewUsername = findViewById(R.id.editTextNewUsername)

editTextNewPassword = findViewById(R.id.editTextNewPassword)

buttonRegisterNew = findViewById(R.id.buttonRegisterNew)

buttonRegisterNew.setOnClickListener {

val newUsername = editTextNewUsername.text.toString()

val newPassword = editTextNewPassword.text.toString()

if (register(newUsername, newPassword)) {

// Registration successful

// You can perform any desired action here

// For simplicity, let's just display a toast message

showToast("Registration successful!")

finish() // Finish the activity and return to the main activity

} else {

// Registration failed

showToast("Registration failed!")

}

}

}

}

private fun showToast(message: String) {

Toast.makeText(this, message, Toast.LENGTH\_SHORT).show()

}

}

**Step7: activity\_profile.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="match\_parent"

android:orientation="vertical"

android:padding="16dp">

<ImageView

android:id="@+id/imageViewProfile"

android:layout\_width="200dp"

android:layout\_height="200dp"

android:src="@drawable/default\_profile\_image" />

<Button

android:id="@+id/buttonUpload"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:text="Upload Profile Picture" />

</LinearLayout>