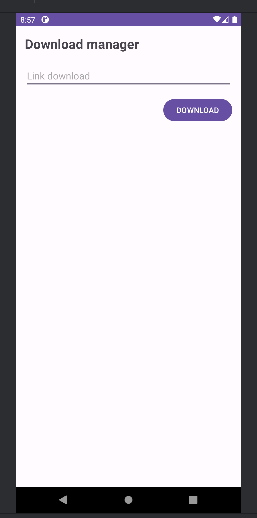
Code 2 file



Code 2 file

DownloadService.java

package com.example.lab7.downloadmanager;  
  
import com.example.lab7.R;  
import android.app.Notification;  
import android.app.NotificationChannel;  
import android.app.NotificationManager;  
import android.app.PendingIntent;  
import android.app.Service;  
import android.content.Context;  
import android.content.Intent;  
import android.os.Build;  
import android.os.IBinder;  
import android.widget.RemoteViews;  
import androidx.annotation.Nullable;  
import androidx.core.app.NotificationCompat;  
  
import java.io.File;  
import java.io.FileOutputStream;  
import java.io.InputStream;  
import java.net.HttpURLConnection;  
import java.net.URL;  
  
public class DownloadService extends Service {  
 private static final int *NOTIFICATION\_ID* = 1;  
 private static final String *CHANNEL\_ID* = "DownloadChannel";  
 public static final String *ACTION\_PAUSE* = "ACTION\_PAUSE";  
 public static final String *ACTION\_RESUME* = "ACTION\_RESUME";  
 public static final String *ACTION\_CANCEL* = "ACTION\_CANCEL";  
  
 private volatile boolean isPaused = false;  
 private volatile boolean isCancelled = false;  
 private Thread downloadThread;  
  
 @Override  
 public void onCreate() {  
 super.onCreate();  
 createNotificationChannel();  
 }  
  
 @Override  
 public int onStartCommand(Intent intent, int flags, int startId) {  
 String urlString = intent.getStringExtra("url");  
  
 if (urlString != null) {  
 isPaused = false;https://www.google.com/  
 isCancelled = false;  
 startForeground(*NOTIFICATION\_ID*, createNotification("Starting download...", urlString, 0));  
 // Bắt đầu download trên một thread riêng  
 downloadThread = new Thread(() -> startDownload(urlString));  
 downloadThread.start();  
 } else if (intent.getAction() != null) {  
 handleAction(intent.getAction());  
 }  
  
 return *START\_NOT\_STICKY*;  
 }  
  
 private void handleAction(String action) {  
 switch (action) {  
 case *ACTION\_PAUSE*:  
 isPaused = true;  
 break;  
 case *ACTION\_RESUME*:  
 isPaused = false;  
 synchronized (downloadThread) {  
 downloadThread.notify();  
 }  
 break;  
 case *ACTION\_CANCEL*:  
 isCancelled = true;  
 if (downloadThread != null) {  
 downloadThread.interrupt();  
 }  
 stopForeground(true);  
 stopSelf();  
 break;  
 }  
 }  
  
  
 private void startDownload(String urlString) {  
 HttpURLConnection connection = null;  
 InputStream input = null;  
 FileOutputStream output = null;  
  
 try {  
 URL url = new URL(urlString);  
 connection = (HttpURLConnection) url.openConnection();  
 connection.connect();  
  
 if (connection.getResponseCode() != HttpURLConnection.*HTTP\_OK*) {  
 // Xử lý lỗi server  
 return;  
 }  
  
 int fileLength = connection.getContentLength();  
 input = connection.getInputStream();  
 // Lấy tên file từ URL  
 String fileName = urlString.substring(urlString.lastIndexOf('/') + 1);  
 File outputFile = new File(getFilesDir(), fileName);  
 output = new FileOutputStream(outputFile);  
  
  
 byte[] data = new byte[4096];  
 long total = 0;  
 int count;  
  
 while ((count = input.read(data)) != -1) {  
 if (isCancelled) {  
 // Người dùng đã hủy  
 output.close();  
 input.close();  
 outputFile.delete(); // Xóa file tải dở  
 return;  
 }  
  
 while (isPaused) {  
 updateNotification("Paused", urlString, (int) (total \* 100 / fileLength), true);  
 synchronized (downloadThread) {  
 try {  
 downloadThread.wait();  
 } catch (InterruptedException e) {  
 Thread.*currentThread*().interrupt();  
 return;  
 }  
 }  
 }  
  
 total += count;  
 if (fileLength > 0) {  
 int progress = (int) (total \* 100 / fileLength);  
 updateNotification("Downloading...", urlString, progress, false);  
 }  
 output.write(data, 0, count);  
 }  
  
 // Download hoàn tất  
 stopForeground(true);  
 updateNotification("Download Complete", urlString, 100, false);  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 // Xử lý lỗi download  
 } finally {  
 try {  
 if (output != null) output.close();  
 if (input != null) input.close();  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 if (connection != null) connection.disconnect();  
 }  
 }  
  
 private void updateNotification(String status, String url, int progress, boolean paused) {  
 NotificationManager manager = (NotificationManager) getSystemService(Context.*NOTIFICATION\_SERVICE*);  
 manager.notify(*NOTIFICATION\_ID*, createNotification(status, url, progress, paused));  
 }  
  
  
 private Notification createNotification(String status, String url, int progress, boolean... isPausedOverride) {  
 boolean paused = isPausedOverride.length > 0 && isPausedOverride[0];  
  
 // Custom layout cho notification  
 RemoteViews remoteViews = new RemoteViews(getPackageName(), R.layout.*custom\_notification*);  
 remoteViews.setTextViewText(R.id.*tvTitle*, "Download manager");  
 remoteViews.setTextViewText(R.id.*tvLink*, "Link: " + url);  
 remoteViews.setTextViewText(R.id.*tvProgress*, status + " " + progress + "%");  
  
 if (paused) {  
 remoteViews.setTextViewText(R.id.*btnPauseResume*, "Resume");  
 remoteViews.setOnClickPendingIntent(R.id.*btnPauseResume*, createPendingIntent(*ACTION\_RESUME*));  
 } else {  
 remoteViews.setTextViewText(R.id.*btnPauseResume*, "Pause");  
 remoteViews.setOnClickPendingIntent(R.id.*btnPauseResume*, createPendingIntent(*ACTION\_PAUSE*));  
 }  
  
 remoteViews.setOnClickPendingIntent(R.id.*btnCancel*, createPendingIntent(*ACTION\_CANCEL*));  
  
  
 NotificationCompat.Builder builder = new NotificationCompat.Builder(this, *CHANNEL\_ID*)  
 .setSmallIcon(R.drawable.*ic\_download*) // Bạn cần tạo icon này trong res/drawable  
 .setCustomContentView(remoteViews)  
 .setOnlyAlertOnce(true);  
  
 // Nếu download xong thì remove notification  
 if (progress == 100 && status.contains("Complete")) {  
 builder.setTimeoutAfter(5000); // Tự động xóa sau 5 giây  
 } else {  
 builder.setOngoing(true); // Không cho người dùng vuốt xóa  
 }  
  
  
 return builder.build();  
 }  
  
  
 private PendingIntent createPendingIntent(String action) {  
 Intent intent = new Intent(this, NotificationReceiver.class);  
 intent.setAction(action);  
 return PendingIntent.*getBroadcast*(this, 0, intent, PendingIntent.*FLAG\_UPDATE\_CURRENT* | PendingIntent.*FLAG\_IMMUTABLE*);  
 }  
  
 private void createNotificationChannel() {  
 if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*O*) {  
 NotificationChannel serviceChannel = new NotificationChannel(  
 *CHANNEL\_ID*,  
 "Download Service Channel",  
 NotificationManager.*IMPORTANCE\_LOW* );  
 NotificationManager manager = getSystemService(NotificationManager.class);  
 manager.createNotificationChannel(serviceChannel);  
 }  
 }  
  
 @Nullable  
 @Override  
 public IBinder onBind(Intent intent) {  
 return null;  
 }  
}

Notification recevicer

package com.example.lab7.downloadmanager;  
  
import android.content.BroadcastReceiver;  
import android.content.Context;  
import android.content.Intent;  
  
public class NotificationReceiver extends BroadcastReceiver {  
 @Override  
 public void onReceive(Context context, Intent intent) {  
 // Gửi Intent đến Service để xử lý  
 Intent serviceIntent = new Intent(context, DownloadService.class);  
 serviceIntent.setAction(intent.getAction());  
 context.startService(serviceIntent);  
 }  
}