

MainActivity.java

package com.example.bai7;  
  
import android.Manifest;  
import android.content.Intent;  
import android.content.pm.PackageManager;  
import android.os.Build;  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.Toast;  
  
import androidx.activity.result.ActivityResultLauncher;  
import androidx.activity.result.contract.ActivityResultContracts;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.core.content.ContextCompat;  
  
import com.google.android.material.textfield.TextInputEditText;  
  
public class MainActivity extends AppCompatActivity {  
  
 private TextInputEditText edtLinkDownload;  
 private Button btnDownload;  
  
 private final ActivityResultLauncher<String> requestPermissionLauncher =  
 registerForActivityResult(new ActivityResultContracts.RequestPermission(), isGranted -> {  
 if (!isGranted) {  
 Toast.*makeText*(this, "Quyền gửi thông báo bị từ chối!", Toast.*LENGTH\_SHORT*).show();  
 }  
 });  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 edtLinkDownload = findViewById(R.id.*edt\_link\_download*);  
 btnDownload = findViewById(R.id.*btn\_download*);  
  
 askNotificationPermission();  
  
 btnDownload.setOnClickListener(v -> {  
 String url = edtLinkDownload.getText().toString().trim();  
 if (url.isEmpty() || !android.util.Patterns.*WEB\_URL*.matcher(url).matches()) {  
 Toast.*makeText*(MainActivity.this, "Vui lòng nhập một URL hợp lệ", Toast.*LENGTH\_SHORT*).show();  
 return;  
 }  
 startDownload(url);  
 });  
 }  
  
 private void startDownload(String url) {  
 Intent serviceIntent = new Intent(this, DownloadService.class);  
 // \*\*QUAN TRỌNG\*\*: Gửi action để Service biết phải làm gì  
 serviceIntent.setAction(DownloadService.*ACTION\_START\_DOWNLOAD*);  
 serviceIntent.putExtra("url", url);  
  
 ContextCompat.*startForegroundService*(this, serviceIntent);  
 Toast.*makeText*(this, "Bắt đầu tải xuống...", Toast.*LENGTH\_SHORT*).show();  
 }  
  
 private void askNotificationPermission() {  
 if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*TIRAMISU*) {  
 if (ContextCompat.*checkSelfPermission*(this, Manifest.permission.*POST\_NOTIFICATIONS*) !=  
 PackageManager.*PERMISSION\_GRANTED*) {  
 requestPermissionLauncher.launch(Manifest.permission.*POST\_NOTIFICATIONS*);  
 }  
 }  
 }  
}

DownloadService.java

package com.example.bai7;  
  
import android.app.Notification;  
import android.app.NotificationChannel;  
import android.app.NotificationManager;  
import android.app.PendingIntent;  
import android.app.Service;  
import android.content.Intent;  
import android.os.Build;  
import android.os.IBinder;  
import android.util.Log;  
import android.view.View;  
import android.widget.RemoteViews;  
import androidx.annotation.Nullable;  
import androidx.core.app.NotificationCompat;  
  
import java.io.File;  
import java.io.InputStream;  
import java.io.RandomAccessFile;  
import java.net.HttpURLConnection;  
import java.net.URL;  
  
public class DownloadService extends Service {  
  
 private static final String *TAG* = "DownloadService";  
 private static final String *CHANNEL\_ID* = "DownloadChannel";  
 private static final int *NOTIFICATION\_ID* = 1;  
  
 public static final String *ACTION\_START\_DOWNLOAD* = "com.example.bai7.ACTION\_START\_DOWNLOAD";  
 public static final String *ACTION\_PAUSE* = "com.example.bai7.ACTION\_PAUSE";  
 public static final String *ACTION\_RESUME* = "com.example.bai7.ACTION\_RESUME";  
 public static final String *ACTION\_CANCEL* = "com.example.bai7.ACTION\_CANCEL";  
 private static String *currentDownloadUrl*; // Lưu lại URL để sử dụng khi pause/resume  
  
 private Thread downloadThread;  
 private NotificationManager notificationManager;  
  
 private volatile boolean isPaused = false;  
 private volatile boolean isCancelled = false;  
 private long totalBytesDownloaded = 0;  
  
 @Override  
 public void onCreate() {  
 super.onCreate();  
 notificationManager = getSystemService(NotificationManager.class);  
 createNotificationChannel();  
 }  
  
 @Override  
 public int onStartCommand(Intent intent, int flags, int startId) {  
 if (intent == null || intent.getAction() == null) {  
 return *START\_NOT\_STICKY*;  
 }  
  
 String action = intent.getAction();  
 Log.*d*(*TAG*, "onStartCommand received action: " + action);  
  
 switch (action) {  
 case *ACTION\_START\_DOWNLOAD*:  
 final String url = intent.getStringExtra("url");  
 *currentDownloadUrl* = url; // Lưu lại URL  
 startNewDownload(url);  
 break;  
 case *ACTION\_PAUSE*:  
 pauseDownload();  
 break;  
 case *ACTION\_RESUME*:  
 resumeDownload();  
 break;  
 case *ACTION\_CANCEL*:  
 cancelDownload();  
 break;  
 }  
  
 return *START\_NOT\_STICKY*;  
 }  
  
 private void startNewDownload(String fileUrl) {  
 isPaused = false;  
 isCancelled = false;  
 totalBytesDownloaded = 0;  
  
 // \*\*SỬA LỖI QUAN TRỌNG\*\*: Gọi startForeground NGAY LẬP TỨC với một thông báo tạm thời.  
 Notification initialNotification = createNotification("Đang chuẩn bị tải...", 0, false);  
 startForeground(*NOTIFICATION\_ID*, initialNotification);  
  
 // Sau đó mới bắt đầu luồng tải file  
 downloadThread = new Thread(() -> downloadFile(fileUrl));  
 downloadThread.start();  
 }  
  
 // Các hàm pause, resume, cancel  
 private void pauseDownload() {  
 isPaused = true;  
 Log.*d*(*TAG*, "Download paused");  
 }  
  
 private void resumeDownload() {  
 isPaused = false;  
 synchronized (downloadThread) {  
 downloadThread.notifyAll();  
 }  
 Log.*d*(*TAG*, "Download resumed");  
 }  
  
 private void cancelDownload() {  
 isCancelled = true;  
 if (downloadThread != null) {  
 downloadThread.interrupt();  
 }  
 // stopForeground(true) và stopSelf() sẽ được gọi trong khối finally của luồng download  
 }  
  
 private void downloadFile(String fileUrl) {  
 RandomAccessFile randomAccessFile = null;  
 InputStream inputStream = null;  
 HttpURLConnection connection = null;  
  
 try {  
 String fileName = fileUrl.substring(fileUrl.lastIndexOf('/') + 1);  
 File outputFile = new File(getFilesDir(), fileName);  
  
 URL url = new URL(fileUrl);  
 connection = (HttpURLConnection) url.openConnection();  
 connection.setRequestProperty("Range", "bytes=" + totalBytesDownloaded + "-");  
 connection.connect();  
  
 int responseCode = connection.getResponseCode();  
 if (responseCode != HttpURLConnection.*HTTP\_OK* && responseCode != 206 /\* HTTP\_PARTIAL\_CONTENT \*/) {  
 throw new Exception("Server error: " + responseCode);  
 }  
  
 long fileLength = connection.getContentLengthLong();  
 if (responseCode == HttpURLConnection.*HTTP\_OK*) {  
 totalBytesDownloaded = 0; // Download từ đầu  
 } else {  
 fileLength += totalBytesDownloaded; // Resume, cộng thêm phần đã tải  
 }  
  
 randomAccessFile = new RandomAccessFile(outputFile, "rw");  
 randomAccessFile.seek(totalBytesDownloaded);  
 inputStream = connection.getInputStream();  
 byte[] buffer = new byte[4096];  
 int bytesRead;  
  
 while ((bytesRead = inputStream.read(buffer)) != -1) {  
 if (isCancelled) break;  
  
 synchronized (downloadThread) {  
 if (isPaused) {  
 // Cập nhật UI khi đang tạm dừng  
 Notification pauseNotification = createNotification("Đã tạm dừng", (int) (totalBytesDownloaded \* 100 / fileLength), true);  
 notificationManager.notify(*NOTIFICATION\_ID*, pauseNotification);  
 downloadThread.wait(); // Luồng sẽ ngủ ở đây  
 }  
 }  
  
 if (isCancelled) break;  
  
 randomAccessFile.write(buffer, 0, bytesRead);  
 totalBytesDownloaded += bytesRead;  
  
 int progress = (int) (totalBytesDownloaded \* 100 / fileLength);  
 Notification progressNotification = createNotification(progress + "%", progress, false);  
 notificationManager.notify(*NOTIFICATION\_ID*, progressNotification);  
 }  
  
 if (!isCancelled) {  
 Log.*d*(*TAG*, "File downloaded to: " + outputFile.getAbsolutePath());  
 // Hiển thị thông báo thành công  
 NotificationCompat.Builder finalBuilder = new NotificationCompat.Builder(this, *CHANNEL\_ID*)  
 .setSmallIcon(R.drawable.*ic\_launcher\_foreground*)  
 .setContentTitle("Tải xuống hoàn tất")  
 .setContentText(fileName);  
 notificationManager.notify(*NOTIFICATION\_ID*, finalBuilder.build());  
 }  
  
 } catch (Exception e) {  
 if (!isCancelled) {  
 Log.*e*(*TAG*, "Download error", e);  
 Notification errorNotification = createNotification("Tải xuống thất bại", 0, false);  
 notificationManager.notify(*NOTIFICATION\_ID*, errorNotification);  
 }  
 } finally {  
 // Dọn dẹp tài nguyên  
 try {  
 if (randomAccessFile != null) randomAccessFile.close();  
 if (inputStream != null) inputStream.close();  
 if (connection != null) connection.disconnect();  
 } catch (Exception e) { e.printStackTrace(); }  
  
 // Dừng service  
 stopForeground(true);  
 stopSelf();  
 }  
 }  
  
 private Notification createNotification(String contentText, int progress, boolean isPausedState) {  
 String fileName = *currentDownloadUrl* != null ? *currentDownloadUrl*.substring(*currentDownloadUrl*.lastIndexOf('/') + 1) : "Download";  
  
 RemoteViews remoteViews = new RemoteViews(getPackageName(), R.layout.*custom\_notification*);  
 remoteViews.setTextViewText(R.id.*notification\_title*, fileName);  
 remoteViews.setTextViewText(R.id.*notification\_progress\_text*, contentText);  
 remoteViews.setProgressBar(R.id.*notification\_progressbar*, 100, progress, false);  
  
 // Cấu hình PendingIntent cho các nút  
 if (isPausedState) {  
 remoteViews.setViewVisibility(R.id.*btn\_pause*, View.*GONE*);  
 remoteViews.setViewVisibility(R.id.*btn\_resume*, View.*VISIBLE*);  
 Intent resumeIntent = new Intent(this, DownloadActionReceiver.class).setAction(*ACTION\_RESUME*);  
 PendingIntent resumePendingIntent = PendingIntent.*getBroadcast*(this, 1, resumeIntent, PendingIntent.*FLAG\_UPDATE\_CURRENT* | PendingIntent.*FLAG\_IMMUTABLE*);  
 remoteViews.setOnClickPendingIntent(R.id.*btn\_resume*, resumePendingIntent);  
 } else {  
 remoteViews.setViewVisibility(R.id.*btn\_pause*, View.*VISIBLE*);  
 remoteViews.setViewVisibility(R.id.*btn\_resume*, View.*GONE*);  
 Intent pauseIntent = new Intent(this, DownloadActionReceiver.class).setAction(*ACTION\_PAUSE*);  
 PendingIntent pausePendingIntent = PendingIntent.*getBroadcast*(this, 2, pauseIntent, PendingIntent.*FLAG\_UPDATE\_CURRENT* | PendingIntent.*FLAG\_IMMUTABLE*);  
 remoteViews.setOnClickPendingIntent(R.id.*btn\_pause*, pausePendingIntent);  
 }  
  
 Intent cancelIntent = new Intent(this, DownloadActionReceiver.class).setAction(*ACTION\_CANCEL*);  
 PendingIntent cancelPendingIntent = PendingIntent.*getBroadcast*(this, 3, cancelIntent, PendingIntent.*FLAG\_UPDATE\_CURRENT* | PendingIntent.*FLAG\_IMMUTABLE*);  
 remoteViews.setOnClickPendingIntent(R.id.*btn\_cancel*, cancelPendingIntent);  
  
 return new NotificationCompat.Builder(this, *CHANNEL\_ID*)  
 .setSmallIcon(R.drawable.*ic\_launcher\_foreground*) // Thay bằng icon hợp lệ  
 .setCustomContentView(remoteViews)  
 .setStyle(new NotificationCompat.DecoratedCustomViewStyle())  
 .setOnlyAlertOnce(true)  
 .setOngoing(true)  
 .build();  
 }  
  
 private void createNotificationChannel() {  
 if (Build.VERSION.*SDK\_INT* >= Build.VERSION\_CODES.*O*) {  
 NotificationChannel channel = new NotificationChannel(*CHANNEL\_ID*, "Download Channel", NotificationManager.*IMPORTANCE\_LOW*);  
 notificationManager.createNotificationChannel(channel);  
 }  
 }  
  
 @Nullable @Override public IBinder onBind(Intent intent) { return null; }  
}

DownloadActionReceiver.java

package com.example.bai7;  
  
import android.content.BroadcastReceiver;  
import android.content.Context;  
import android.content.Intent;  
import android.util.Log;  
  
public class DownloadActionReceiver extends BroadcastReceiver {  
  
 private static final String *TAG* = "DownloadActionReceiver";  
  
 @Override  
 public void onReceive(Context context, Intent intent) {  
 if (intent == null || intent.getAction() == null) {  
 return;  
 }  
  
 // Lấy action từ Intent được gửi từ PendingIntent của notification  
 String action = intent.getAction();  
 Log.*d*(*TAG*, "Received action: " + action);  
  
 // Tạo một Intent mới để gửi lệnh đến DownloadService  
 Intent serviceIntent = new Intent(context, DownloadService.class);  
 serviceIntent.setAction(action); // Gửi cùng action đó cho service  
  
 // Khởi chạy service để nó có thể xử lý lệnh  
 // Service sẽ không bị tạo mới nếu đang chạy, mà chỉ nhận lệnh trong onStartCommand  
 context.startService(serviceIntent);  
 }  
}

activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/main"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:padding="16dp"  
 tools:context=".MainActivity">  
  
 <com.google.android.material.textfield.TextInputLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox">  
  
 <com.google.android.material.textfield.TextInputEditText  
 android:id="@+id/edt\_link\_download"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:hint="Link download" />  
 </com.google.android.material.textfield.TextInputLayout>  
  
 <Button  
 android:id="@+id/btn\_download"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="end"  
 android:layout\_marginTop="8dp"  
 android:text="DOWNLOAD" />  
  
</LinearLayout>