LAB07  
  
AndroidManifest.xml  
*<uses-permission android:name="android.permission.INTERNET"/>  
<uses-permission android:name="android.permission.FOREGROUND\_SERVICE"/>  
<uses-permission android:name="android.permission.FOREGROUND\_SERVICE\_DATA\_SYNC"/>  
<uses-permission android:name="android.permission.POST\_NOTIFICATIONS"/>*

*<application  
 android:allowBackup="true"  
 android:dataExtractionRules="@xml/data\_extraction\_rules"  
 android:fullBackupContent="@xml/backup\_rules"  
 android:icon="@mipmap/ic\_launcher"  
 android:label="@string/app\_name"  
 android:roundIcon="@mipmap/ic\_launcher\_round"  
 android:supportsRtl="true"  
 android:theme="@style/Theme.Lab7">  
 <activity  
 android:name=".MainActivity"  
 android:exported="true">  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN"/>  
 <category android:name="android.intent.category.LAUNCHER"/>  
 </intent-filter>  
 </activity>  
  
 <service  
 android:name=".DownloadService"  
 android:exported="false"  
 android:foregroundServiceType="dataSync" />  
  
 <receiver  
 android:name=".ActionReceiver"  
 android:exported="true">  
 <intent-filter>  
 <action android:name="dm.ACTION\_PAUSE"/>  
 <action android:name="dm.ACTION\_RESUME"/>  
 <action android:name="dm.ACTION\_CANCEL"/>  
 </intent-filter>  
 </receiver>  
</application>*

DownloadService.java

public class DownloadService extends Service {  
 private volatile boolean isPaused = false;  
 private volatile boolean isCanceled = false;  
 private long downloaded = 0L;  
 private long totalSize = -1L;  
 private String urlStr;  
 private String fileName;  
 private File outFile;  
  
 @Override public void onCreate() {  
 super.onCreate();  
 createChannel();  
 }  
  
 @Override public int onStartCommand(Intent intent, int flags, int startId) {  
 String action = intent != null ? intent.getAction() : null;  
  
 if (DM.ACTION\_START.equals(action)) {  
 urlStr = intent.getStringExtra(DM.EXTRA\_URL);  
 fileName = intent.getStringExtra(DM.EXTRA\_NAME);  
 if (fileName == null || fileName.trim().isEmpty()) {  
 fileName = "download.bin";  
 }  
 outFile = new File(getFilesDir(), fileName); // internal storage  
 isPaused = false; isCanceled = false;  
  
 startForeground(DM.NOTI\_ID, buildNoti(0, "Bắt đầu tải..."));  
 new Thread(this::downloadLoop).start();  
 } else if (DM.ACTION\_PAUSE.equals(action)) {  
 isPaused = true;  
 updateNoti(-1, "Đã tạm dừng");  
 } else if (DM.ACTION\_RESUME.equals(action)) {  
 if (isPaused) {  
 isPaused = false;  
 new Thread(this::downloadLoop).start();  
 updateNoti(-1, "Tiếp tục tải...");  
 }  
 } else if (DM.ACTION\_CANCEL.equals(action)) {  
 isCanceled = true;  
 stopSelf();  
 }  
 return START\_NOT\_STICKY;  
 }  
  
 private void downloadLoop() {  
 HttpURLConnection conn = null;  
 InputStream in = null;  
 RandomAccessFile raf = null;  
 try {  
 if (outFile.exists()) downloaded = outFile.length();  
  
 URL url = new URL(urlStr);  
 conn = (HttpURLConnection) url.openConnection();  
 conn.setConnectTimeout(15000);  
 conn.setReadTimeout(30000);  
  
 // Resume bằng Range header  
 if (downloaded > 0) {  
 conn.setRequestProperty("Range", "bytes=" + downloaded + "-");  
 }  
 conn.connect();  
  
 int code = conn.getResponseCode();  
 // 206 = partial content khi resume; 200 = tải mới  
 if (code == HttpURLConnection.HTTP\_OK) {  
 totalSize = conn.getContentLengthLong();  
 downloaded = 0;  
 } else if (code == HttpURLConnection.HTTP\_PARTIAL) {  
 long len = conn.getContentLengthLong();  
 totalSize = downloaded + len;  
 } else {  
 updateNoti(-1, "Lỗi HTTP " + code);  
 stopSelf();  
 return;  
 }  
  
 in = new BufferedInputStream(conn.getInputStream());  
 raf = new RandomAccessFile(outFile, "rw");  
 raf.seek(downloaded);  
  
 byte[] buf = new byte[8192];  
 long lastUpdate = SystemClock.uptimeMillis();  
 int n;  
 while ((n = in.read(buf)) != -1) {  
 if (isCanceled) { outFile.delete(); break; }  
 while (isPaused) { SystemClock.sleep(200); if (isCanceled) break; }  
 if (isCanceled) break;  
  
 raf.write(buf, 0, n);  
 downloaded += n;  
  
 long now = SystemClock.uptimeMillis();  
 if (now - lastUpdate > 500) {  
 int progress = totalSize > 0 ? (int) (downloaded \* 100 / totalSize) : 0;  
 updateNoti(progress, "Đang tải...");  
 lastUpdate = now;  
 }  
 }  
  
 if (isCanceled) {  
 updateNoti(-1, "Đã hủy");  
 } else if (!isPaused) {  
 updateNoti(100, "Hoàn tất");  
 }  
 } catch (Exception e) {  
 updateNoti(-1, "Lỗi: " + e.getMessage());  
 } finally {  
 try { if (in != null) in.close(); } catch (IOException ignored) {}  
 try { if (raf != null) raf.close(); } catch (IOException ignored) {}  
 if (!isPaused) stopSelf();  
 if (conn != null) conn.disconnect();  
 }  
 }  
  
 private Notification buildNoti(int progress, String status) {  
 RemoteViews rv = new RemoteViews(getPackageName(), R.layout.notify\_download);  
 rv.setTextViewText(R.id.txtTitle, status);  
 if (progress >= 0) {  
 rv.setProgressBar(R.id.progress, 100, progress, false);  
 }  
  
 // PendingIntents cho 3 nút  
 rv.setOnClickPendingIntent(R.id.btnPause, pi(this, DM.ACTION\_PAUSE));  
 rv.setOnClickPendingIntent(R.id.btnResume, pi(this, DM.ACTION\_RESUME));  
 rv.setOnClickPendingIntent(R.id.btnCancel, pi(this, DM.ACTION\_CANCEL));  
  
 NotificationCompat.Builder b = new NotificationCompat.Builder(this, DM.CH\_ID)  
 .setSmallIcon(android.R.drawable.stat\_sys\_download)  
 .setContent(rv)  
 .setOngoing(true)  
 .setOnlyAlertOnce(true)  
 .setLargeIcon(BitmapFactory.decodeResource(getResources(), android.R.drawable.stat\_sys\_download));  
  
 return b.build();  
 }  
  
 private void updateNoti(int progress, String status) {  
 NotificationManager nm = (NotificationManager) getSystemService(NOTIFICATION\_SERVICE);  
 nm.notify(DM.NOTI\_ID, buildNoti(progress, status));  
 }  
  
 private PendingIntent pi(Context c, String action) {  
 Intent i = new Intent(c, ActionReceiver.class).setAction(action);  
 int flags = PendingIntent.FLAG\_UPDATE\_CURRENT | PendingIntent.FLAG\_IMMUTABLE;  
 return PendingIntent.getBroadcast(c, action.hashCode(), i, flags);  
 }  
  
 private void createChannel() {  
 if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.O) {  
 NotificationChannel ch = new NotificationChannel(  
 DM.CH\_ID, "Download Manager", NotificationManager.IMPORTANCE\_LOW);  
 ((NotificationManager) getSystemService(NOTIFICATION\_SERVICE)).createNotificationChannel(ch);  
 }  
 }  
  
 @Nullable @Override public IBinder onBind(Intent intent) { return null; }  
}

ActionReceiver.java

public class ActionReceiver extends BroadcastReceiver {  
 @Override public void onReceive(Context ctx, Intent i) {  
 String action = i.getAction();  
 Intent svc = new Intent(ctx, DownloadService.class);  
 if (DM.ACTION\_PAUSE.equals(action)) {  
 svc.setAction(DM.ACTION\_PAUSE);  
 } else if (DM.ACTION\_RESUME.equals(action)) {  
 svc.setAction(DM.ACTION\_RESUME);  
 } else if (DM.ACTION\_CANCEL.equals(action)) {  
 svc.setAction(DM.ACTION\_CANCEL);  
 }  
 ctx.startService(svc);  
 }  
}

MainActivity.java

public class MainActivity extends AppCompatActivity {  
 EditText edtUrl;  
 Button btnDownload, btnPause, btnResume, btnCancel;  
  
 ActivityResultLauncher<String> reqNoti = registerForActivityResult(  
 new ActivityResultContracts.RequestPermission(), granted -> {}  
 );  
  
 @Override protected void onCreate(@Nullable Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main);  
  
 edtUrl = findViewById(R.id.edtUrl);  
 btnDownload = findViewById(R.id.btnDownload);  
 btnPause = findViewById(R.id.btnPause);  
 btnResume = findViewById(R.id.btnResume);  
 btnCancel = findViewById(R.id.btnCancel);  
  
 askNotificationIfNeeded();  
  
 btnDownload.setOnClickListener(v -> {  
 Toast.makeText(this, "CLICKED", Toast.LENGTH\_SHORT).show();  
 android.util.Log.d("DM", "btnDownload clicked");  
 // Start demo service  
 Intent i = new Intent(this, DemoFgService.class);  
 i.setAction("demo.START");  
 androidx.core.content.ContextCompat.startForegroundService(this, i);  
 });  
  
 btnPause.setOnClickListener(v -> sendAction(DM.ACTION\_PAUSE));  
 btnResume.setOnClickListener(v -> sendAction(DM.ACTION\_RESUME));  
 btnCancel.setOnClickListener(v -> sendAction(DM.ACTION\_CANCEL));  
 }  
  
 private void sendAction(String action) {  
 Intent i = new Intent(this, DownloadService.class);  
 i.setAction(action);  
 startService(i);  
 }  
  
 private String guessName(String url) {  
 int p = url.lastIndexOf('/');  
 return (p >= 0 && p < url.length()-1) ? url.substring(p+1) : "download.bin";  
 }  
  
 private void askNotificationIfNeeded() {  
 if (Build.VERSION.SDK\_INT >= 33) {  
 if (ContextCompat.checkSelfPermission(this, Manifest.permission.POST\_NOTIFICATIONS)  
 != PackageManager.PERMISSION\_GRANTED) {  
 reqNoti.launch(Manifest.permission.POST\_NOTIFICATIONS);  
 }  
 }  
 }

Activity\_main.xml

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:padding="16dp" android:orientation="vertical"  
 android:layout\_width="match\_parent" android:layout\_height="match\_parent">  
  
 <EditText  
 android:id="@+id/edtUrl"  
 android:hint="Paste link cần download..."  
 android:inputType="textUri"  
 android:layout\_width="match\_parent" android:layout\_height="wrap\_content"/>  
  
 <Button android:id="@+id/btnDownload" android:text="Download"  
 android:layout\_width="match\_parent" android:layout\_height="wrap\_content" />  
  
 <LinearLayout android:orientation="horizontal"  
 android:layout\_width="match\_parent" android:layout\_height="wrap\_content"  
 android:gravity="center" android:paddingTop="8dp">  
 <Button android:id="@+id/btnPause" android:text="Pause"  
 android:layout\_width="0dp" android:layout\_weight="1"  
 android:layout\_height="wrap\_content"/>  
 <Button android:id="@+id/btnResume" android:text="Resume"  
 android:layout\_width="0dp" android:layout\_weight="1"  
 android:layout\_height="wrap\_content"/>  
 <Button android:id="@+id/btnCancel" android:text="Cancel"  
 android:layout\_width="0dp" android:layout\_weight="1"  
 android:layout\_height="wrap\_content"/>  
 </LinearLayout>  
</LinearLayout>

Notify\_download.xml

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:padding="12dp" android:layout\_width="match\_parent" android:layout\_height="wrap\_content">  
 <TextView android:id="@+id/txtTitle" android:text="Đang tải..."  
 android:textStyle="bold" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"/>  
 <ProgressBar android:id="@+id/progress" style="?android:attr/progressBarStyleHorizontal"  
 android:max="100" android:progress="0"  
 android:layout\_below="@id/txtTitle"  
 android:layout\_width="match\_parent" android:layout\_height="wrap\_content"/>  
 <LinearLayout  
 android:layout\_below="@id/progress" android:layout\_width="match\_parent" android:layout\_height="wrap\_content"  
 android:orientation="horizontal" android:gravity="end" android:paddingTop="6dp">  
 <Button android:id="@+id/btnPause" android:text="Pause" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"/>  
 <Button android:id="@+id/btnResume" android:text="Resume" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"/>  
 <Button android:id="@+id/btnCancel" android:text="Cancel" android:layout\_width="wrap\_content" android:layout\_height="wrap\_content"/>  
 </LinearLayout>  
</RelativeLayout>

DEMO

A screenshot of a phone

AI-generated content may be incorrect.