

# UNIVERSIDAD CONTINENTAL

FACULTAD DE INGENIERÍA

ESCUELA ACADÉMICO PROFESIONAL DE INGENIERÍA  
DE SISTEMAS E INFORMÁTICA

**CURSO:**

TALLER DE PROYECTOS - I - ING. DE SISTEMAS E  
INFORMÁTICA

**DOCENTE:**

AMERICO ESTRADA SANCHEZ

**TEMA:**

“SafeRoute: Desarrollo de un Sistema Web para  
optimización de rutas seguras en Cusco; 2025”

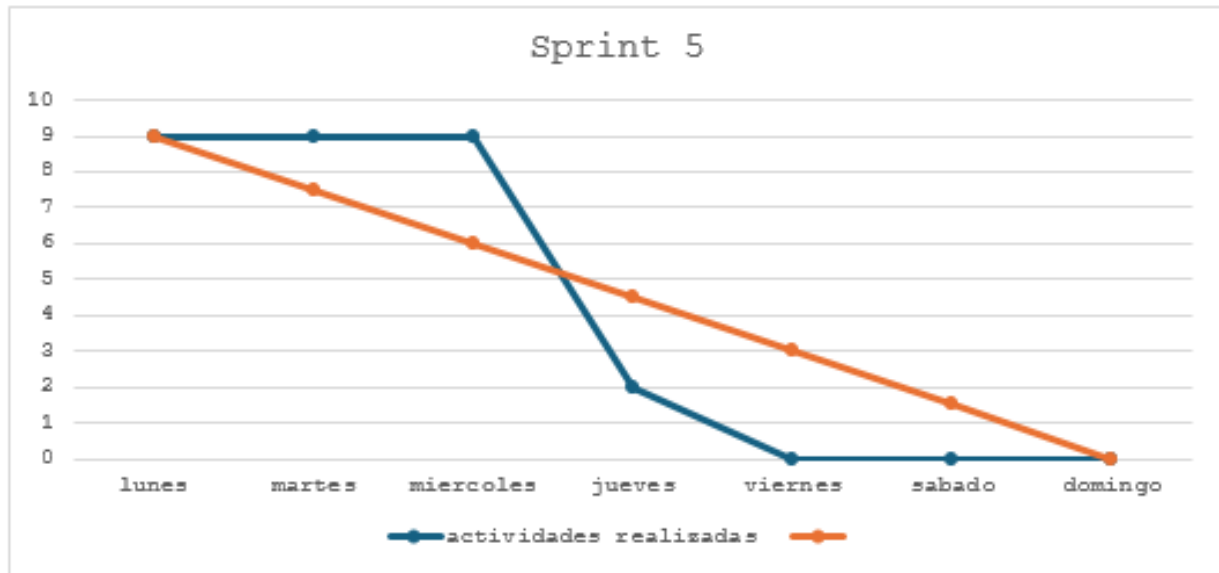
**PRESENTADO POR:**

APELLIDOS Y NOMBRES	CÓDIGO
CASTILLO CCANTO FRANK	74534208
ESPETIA MAMANI JHON CRISTIAN	74988478
RAFAELE HUAMAN LUIS CRISTIAN	74653997

## SPRINT 5

#	HU	Actividad	Tiempo(h)
48	HU-23	Implementar opción compartir (WhatsApp/SMS)	1.50
49	HU-23	Enviar ubicación actualizada periódicamente	2.50
50	HU-24	Guardar rutas en BD	3.00
51	HU-24	Mostrar historial con fecha/hora	2.65
52	HU-13	Crear formulario	2.00
53	HU-13	Guardar en BD	2.00
54	HU-13	Asociar coordenadas	1.65
49	HU-19	Configurar notificaciones	4.00
50	HU-19	Asociar a nivel de riesgo	2.50
<b>Total</b>			<b>20.76</b>

## BURN DOWN CHART



Nº	Actividades - HU	Nombre	Por hacer	Haciendo	Hecho
1	HU-23	Implementar opción compartir (WhatsApp/SMS)			Implementar opción compartir (WhatsApp/SMS)
2	HU-23	Enviar ubicación actualizada periódicamente			Enviar ubicación actualizada periódicamente
3	HU-24	Guardar rutas en BD			Guardar rutas en BD
4	HU-24	Mostrar historial con fecha/hora			Guardar rutas en BD
5	HU-13	Crear formulario			Crear formulario
6	HU-13	Guardar en BD			Guardar en BD
7	HU-13	Asociar coordenadas			Asociar coordenadas
8	HU-10	Configurar notificaciones			Configurar notificaciones
9	HU-10	Asociar a nivel de riesgo			Asociar a nivel de riesgo

## Documentacion sprint-5

#	HU	Actividad	Tiempo(h)
48	HU-23	Implementar opción compartir (WhatsApp/SMS)	1.50
49	HU-23	Enviar ubicación actualizada periódicamente	2.50
50	HU-24	Guardar rutas en BD	3.00

### a) implementar opcion compartir whatsapp

```
btnCompartir.setOnClickListener( View v -> {  
    if (origenLatLng != null && destinoLatLng != null) {  
        // Generar enlace de Google Maps con ruta  
        String urlMaps = "https://www.google.com/maps/dir/?api=1" +  
            "&origin=" + origenLatLng.latitude + "," + origenLatLng.longitude +  
            "&destination=" + destinoLatLng.latitude + "," + destinoLatLng.longitude +  
            "&travelmode=" + modoTransporte; // driving, walking, bicycling  
  
        // Crear intent para WhatsApp (puedes cambiar a SMS si quieres)  
        Intent intent = new Intent(Intent.ACTION_SEND);  
        intent.setType("text/plain");  
        intent.setPackage("com.whatsapp"); // eliminar esta linea para usar cualquier app de mensajeria  
        intent.putExtra(Intent.EXTRA_TEXT, "Mira esta ruta: " + urlMaps);  
  
        try {  
            startActivity(intent);  
        } catch (android.content.ActivityNotFoundException ex) {  
            Toast.makeText(requireContext(), "WhatsApp no está instalado.", Toast.LENGTH_SHORT).show();  
        }  
    } else {  
        Toast.makeText(requireContext(), "Selecciona origen y destino primero", Toast.LENGTH_SHORT).show();  
    }  
});
```


☰

Mapa


⋮

Español

▼



Plaza Túpac Amaru, Cusco, Perú



Molino, Cusco, Perú

Carro

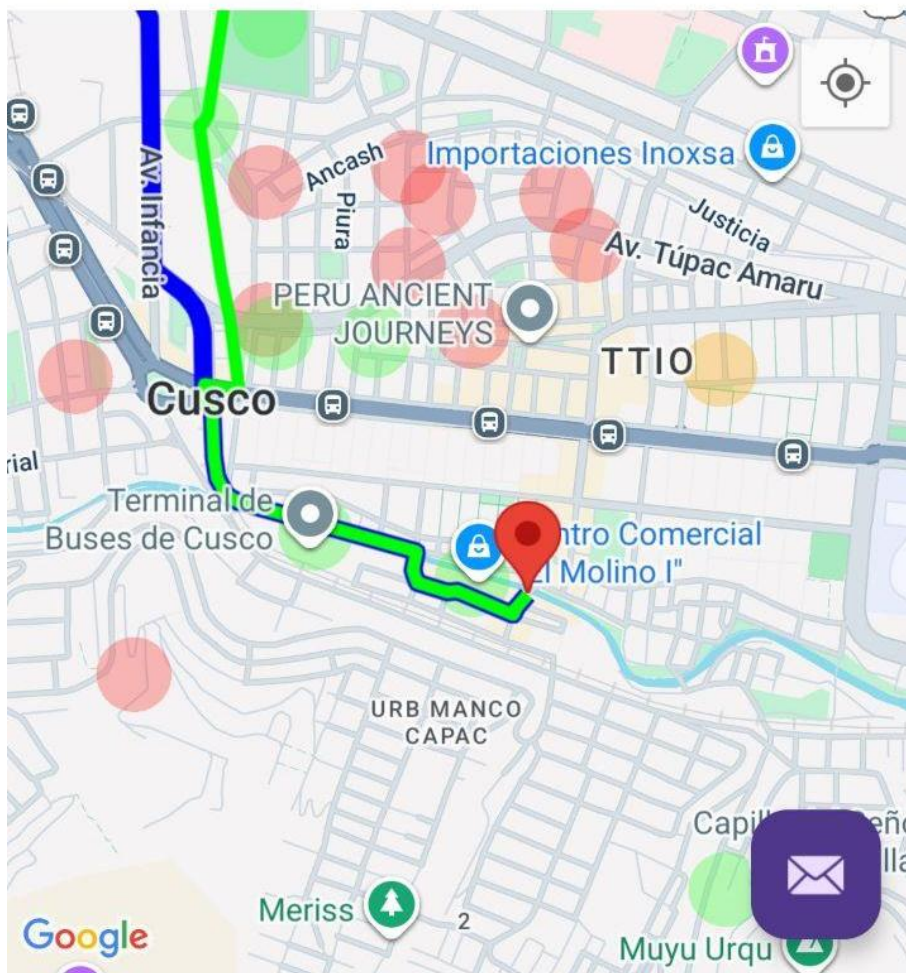
▼

Calcular Ruta

Guardar Ruta

Compartir Ubi.

Comparando | Azul: Corta (10 min) | Verde: Segura (9 min)



## b) enviar ubicacion actualizada periodicamente

```
private void iniciarEnvioUbicacion() {
    runnable = new Runnable() {
        @Override
        public void run() {
            if (ActivityCompat.checkSelfPermission(requireContext(), Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
                ActivityCompat.requestPermissions(requireActivity(), new String[]{Manifest.permission.ACCESS_FINE_LOCATION}, 1);
                return;
            }

            fusedLocationClient.getLastLocation().addOnSuccessListener( Location location -> {
                if (location != null) {
                    // Enviar ubicación actual
                    String mensaje = "Ubicación actual: " + location.getLatitude() + "," + location.getLongitude();

                    // Aquí puedes enviar vía Intent, Firestore, o lo que necesites
                    Toast.makeText(requireContext(), "Enviando: " + mensaje, Toast.LENGTH_SHORT).show();
                }
            });

            handler.postDelayed(this, intervaloEnvio);
        }
    };
    handler.post(runnable);
}
```

Plaza Túpac Amaru, Cusco, Perú

Molino, Cusco, Perú

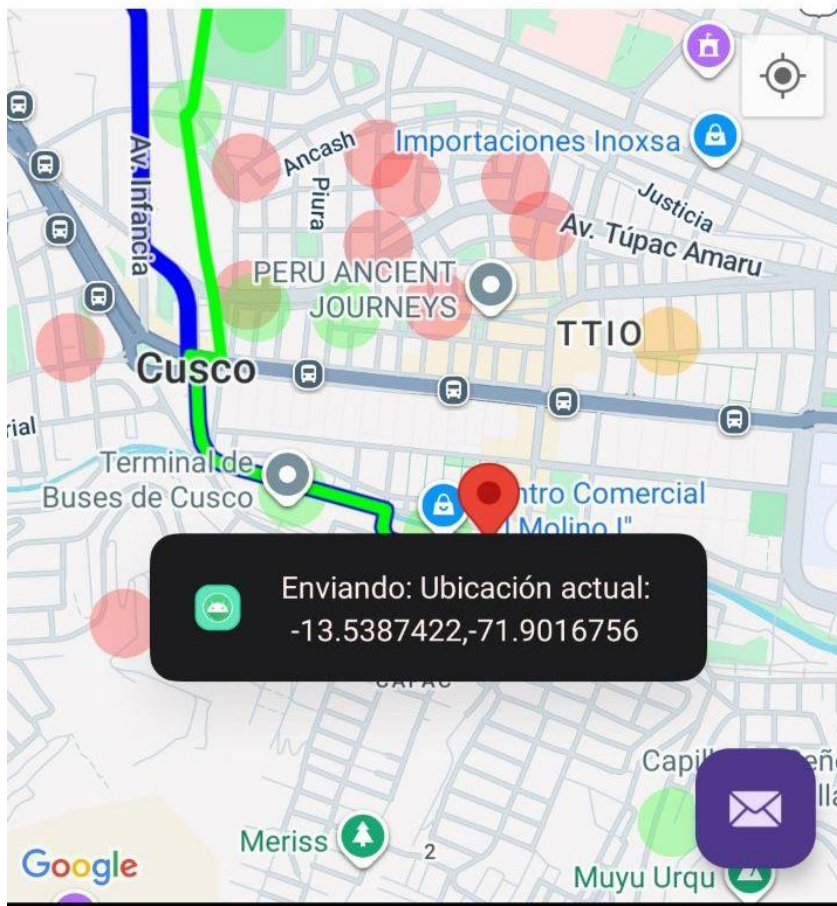
Carro

Calcular Ruta

Guardar Ruta

Compartir Ubi.

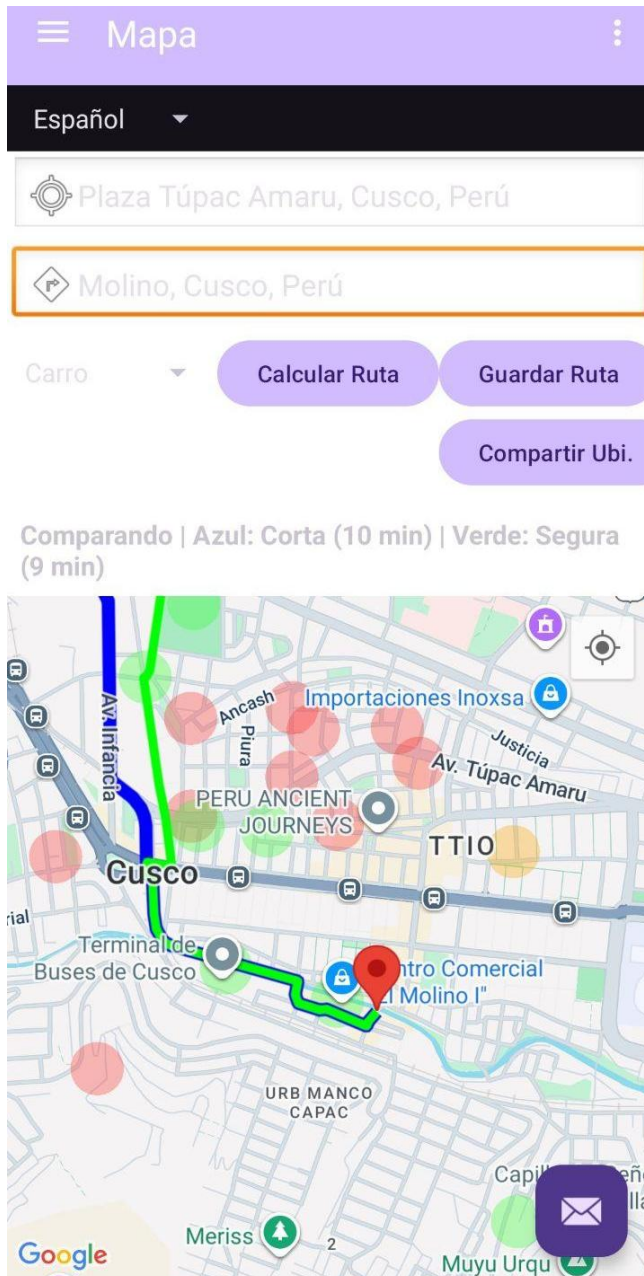
Comparando | Azul: Corta (10 min) | Verde: Segura (9 min)





### c) Guardar ruta en BD

```
btnGuardarRuta.setOnClickListener( View v -> {  
    if (origenLatIng != null && destinoLatIng != null) {  
        // Aquí puedes guardar la ruta en Firestore o local  
        Toast.makeText(requireContext(), "Ruta guardada exitosamente", Toast.LENGTH_SHORT).show();  
    } else {  
        Toast.makeText(requireContext(), "Selecciona origen y destino primero", Toast.LENGTH_SHORT).show();  
    }  
});
```





## SPRINT 5

Mostrar historial de rutas
Crear formulario

la implementación de la característica "Historial de Rutas". El objetivo es permitir que los usuarios guarden las rutas que calculan y puedan consultarlas más tarde. Al hacer clic en una ruta guardada en el historial, el usuario es redirigido al mapa, donde se traza automáticamente el trayecto seleccionado.

### A. El Modelo de Datos: `HistorialRuta.java`

Este archivo es la "plantilla" o POJO (Plain Old Java Object) que define la estructura de los datos que se guardarán en la colección `historialRutas` de Firestore.

```
package com.example.myapplicationf.Models;

import com.google.firebase.Timestamp;
import java.util.Date;

public class HistorialRuta {

    private String userId;

    private String origenNombre;

    private String destinoNombre;

    private double origenLat;

    private double origenLng;

    private double destinoLat;

    private double destinoLng;

    private Timestamp timestamp;

    // Constructor vacío requerido por Firestore

    public HistorialRuta() {}
```

```
// Constructor completo

public HistorialRuta(String userId, String origenNombre, String
destinoNombre,

                        double origenLat, double origenLng, double
destinoLat, double destinoLng,

                        Timestamp timestamp) {

    this.userId = userId;

    this.origenNombre = origenNombre;

    this.destinoNombre = destinoNombre;

    this.origenLat = origenLat;

    this.origenLng = origenLng;

    this.destinoLat = destinoLat;

    this.destinoLng = destinoLng;

    this.timestamp = timestamp;

}

// Getters y Setters para todos los campos

public String getUserId() { return userId; }

public void setUserId(String userId) { this.userId = userId; }

public String getOrigenNombre() { return origenNombre; }

public void setOrigenNombre(String origenNombre) { this.origenNombre =
origenNombre; }

public String getDestinoNombre() { return destinoNombre; }

public void setDestinoNombre(String destinoNombre) { this.destinoNombre
= destinoNombre; }

public double getOrigenLat() { return origenLat; }
```

```

    public void setOrigenLat(double origenLat) { this.origenLat = origenLat;
}

    public double getOrigenLng() { return origenLng; }

    public void setOrigenLng(double origenLng) { this.origenLng = origenLng;
}

    public double getDestinoLat() { return destinoLat; }

    public void setDestinoLat(double destinoLat) { this.destinoLat =
destinoLat; }

    public double getDestinoLng() { return destinoLng; }

    public void setDestinoLng(double destinoLng) { this.destinoLng =
destinoLng; }

    public Timestamp getTimestamp() { return timestamp; }

    public void setTimestamp(Timestamp timestamp) { this.timestamp =
timestamp; }

}

```

## B. La Pantalla del Mapa: HomeFragment.java

1. **Recibir y dibujar una ruta del historial:** Comprueba si recibe datos (coordenadas) al ser creado. Si es así, traza la ruta automáticamente.

### \* Código para Guardar la Ruta en Firestore

Esta funcionalidad se activa cuando el usuario presiona el botón "Guardar Ruta". Se divide en dos partes: el *listener* del botón y el método que hace el trabajo de guardar.

- El `onClickListener` del Botón `btnGuardarRuta`:

Este bloque de código, que se encuentra en el método `onCreateView`, es el que "escucha" cuándo el usuario presiona el botón. Verifica que haya una ruta válida antes de llamar al método que la guardará.

```
btnGuardarRuta.setOnClickListener(v -> {

    if (origenLatLng != null && destinoLatLng != null && origenNombre != null && destinoNombre != null) {

// Si hay una ruta válida, llama al método para guardarla

guardarRutaEnHistorial();

    } else {

        // Si no, muestra un mensaje de advertencia

        Toast.makeText(requireContext(),
getString(R.string.historial_selecciona_ruta), Toast.LENGTH_SHORT).show();

    }

});
```

### - El Método `guardarRutaEnHistorial()`:

Este es el método que se encarga de toda la lógica de guardado. Crea un objeto `HistorialRuta` con todos los datos necesarios (ID del usuario, nombres, coordenadas y fecha/hora) y lo envía a la colección `historialRutas` en tu base de datos de Firestore.

```
private void guardarRutaEnHistorial() {

    // Obtiene el usuario actual para asignarle la ruta

    FirebaseAuth.getInstance().getCurrentUser();

    if (user == null) {

        Toast.makeText(getContext(), getString(R.string.historial_no_login),
Toast.LENGTH_SHORT).show();

        return;

    }

    String uid = user.getId();

    // Crea un nuevo objeto "HistorialRuta" con todos los datos
```

```

HistorialRuta nuevaRuta = new HistorialRuta(

    uid,

    origenNombre,

    destinoNombre,

    origenLatLng.latitude,

    origenLatLng.longitude,

    destinoLatLng.latitude,

    destinoLatLng.longitude,

    new Timestamp(new Date()) // Añade la fecha y hora actual

);

// Guarda el objeto en la colección "historialRutas" de Firestore

db.collection("historialRutas")

    .add(nuevaRuta)

    .addOnSuccessListener(documentReference ->
Toast.makeText(getContext(), getString(R.string.historial_ruta_guardada),
Toast.LENGTH_SHORT).show())

    .addOnFailureListener(e -> Toast.makeText(getContext(),
getString(R.string.historial_error_guardar), Toast.LENGTH_SHORT).show());

}

```

## \* Código para Recibir y Dibujar la Ruta del Historial

Esta es la parte que se activa cuando el usuario hace clic en un elemento del historial y es redirigido al mapa. Se encuentra dentro del método `onMapReady`.

### A. Bloque para Recibir los Datos:

Este código comprueba si el `HomeFragment` recibió un "paquete" de datos (`Bundle`) con las coordenadas de la ruta. Si los encuentra:

1. Extrae las coordenadas y los nombres.
2. Rellena los campos de texto de origen y destino en la pantalla.
3. Simula un clic en el botón "Calcular Ruta" (`btnCalcularRuta.performClick()`) para que la ruta se trace automáticamente.

```

if (getArguments() != null && getArguments().containsKey("origenLat")) {

Bundle args = getArguments();

// Extrae todos los datos del Bundle

origenLatLng = new LatLng(args.getDouble("origenLat"),
args.getDouble("origenLng"));

destinoLatLng = new LatLng(args.getDouble("destinoLat"),
args.getDouble("destinoLng"));

origenNombre = args.getString("origenNombre");

destinoNombre = args.getString("destinoNombre");


// Rellena los campos de texto para que el usuario vea la ruta seleccionada

etOrigen.setText(origenNombre);

etDestino.setText(destinoNombre);


// Dibujar la ruta automáticamente después de un breve retraso

new Handler().postDelayed(() -> {

    if (mMap != null) {

        // Simula un clic en el botón para trazar la ruta

        btnCalcularRuta.performClick();

    }

    }, 500); // 500ms de espera para asegurar que el mapa esté
listo

}

```

## C. El Diseño de la Lista: `item_historial_ruta.xml` y `fragment_slideshow.xml`

Estos archivos definen la apariencia de la pantalla de historial.

### `item_historial_ruta.xml`

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

```
<androidx.cardview.widget.CardView
xmlns:android="http://schemas.android.com/apk/res/android" (http://schemas.
android.com/apk/res/android) "

xmlns:app="http://schemas.android.com/apk/res-
auto" (http://schemas.android.com/apk/res-auto) "

android:layout_width="match_parent"

android:layout_height="wrap_content"

android:layout_marginHorizontal="8dp"

android:layout_marginVertical="4dp"

app:cardCornerRadius="8dp"

app:cardElevation="2dp">

    <LinearLayout

        android:layout_width="match_parent"

        android:layout_height="wrap_content"

        android:orientation="vertical"

        android:padding="16dp">

            <TextView

                android:id="@+id/tvOrigenHistorial"

                android:layout_width="wrap_content"

                android:layout_height="wrap_content"

                android:textStyle="bold"

                android:textSize="16sp"

                android:textColor="@android:color/black"/>

            <TextView

                android:id="@+id/tvDestinoHistorial"

                android:layout_width="wrap_content"
```



```

        android:layout_height="wrap_content"

        android:layout_marginTop="4dp"

        android:textSize="16sp"

        android:textColor="@android:color/black"/>

        <TextView

        android:id="@+id/tvFechaHistorial"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:layout_marginTop="12dp"

        android:textSize="12sp"

        android:textColor="@android:color/darker_gray"/>

    </LinearLayout>

</androidx.cardview.widget.CardView>

```

## fragment\_slideshow.xml

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android" (http://schemas.
    android.com/apk/res/android) "
    xmlns:tools="http://schemas.android.com/tools" (http://schemas.android.com/
    tools) "
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ui.slideshow.SlideshowFragment">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerViewHistorial"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:padding="8dp"
        tools:listitem="@layout/item_historial_ruta" />

    <TextView
        android:id="@+id/tvEmptyHistory"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true"

```

```
android:text="@string/historial_vacio"  
android:textSize="18sp"  
android:visibility="gone"  
tools:visibility="visible"/>  
</RelativeLayout>
```