Configurando mi propio Endpoint en mi Servidor

Para cumplir con los requisitos del curso, se seleccionó **Heroku** como plataforma gratuita y sencilla para desplegar aplicaciones backend.

¿Por qué Heroku?

Permite desplegar Web Services de forma rápida.

- Acepta varios lenguajes de programación (Java, Node.js, Python, PHP, etc.).
- Tiene un nivel gratuito (free tier) ideal para pruebas.
- Funciona bien para servicios REST que exponen endpoints a través de HTTP.
- No requiere configuración de servidores físicos ni máquinas virtuales.

El servidor se construyó usando **Node.js + Express**, ya que es uno de los lenguajes más ligeros para aplicaciones REST y Heroku lo soporta sin complicaciones.
También sirve perfectamente si tú prefieres otro lenguaje (PHP, Python), pero Express es el estándar más simple.

Ahora bien, el curso pide crear un endpoint llamado:

/registrar-usuario

Este endpoint debe recibir dos parámetros:

- id_dispositivo → el token FCM que genera Firebase en el dispositivo.
- id_usuario_instagram → el usuario configurado en Petagram.

Ambos datos se deben almacenar en una base de datos para poder enviar notificaciones personalizadas en el futuro.

A continuación, se explica paso a paso qué ocurre cuando la app Android envía los datos al servidor:

Recepción de datos desde Android

La aplicación Petagram, cuando el usuario presiona “Recibir Notificaciones”, obtiene el **token FCM** mediante FirebaseMessaging:

Ejemplo:

eTj34sdf8asJH3JHASD... (token)

Luego, la app envía ese token junto con el id del usuario al servidor mediante Retrofit.
Ejemplo del JSON enviado:

{

"id_dispositivo": "eTj34sdf8asJH3JHASD...",
"id_usuario_instagram": "cristian123"
}

Lógica del servidor (backend)

El servidor recibe esos datos y ejecuta:

1. Validación del JSON.
2. Verificación de que ambos campos estén presentes.
3. Conexión a la base de datos.
4. Inserción del registro en la tabla.

Tabla: usuario_instagram

Campo	Tipo	Descripción
id (PK)	Integer	Autonumérico
id_dispositivo	Varchar	Token FCM del celular
id_usuario_instagram	Varchar	Usuario configurado en la app

El backend inserta algo así:

```
INSERT INTO usuario_instagram(id_dispositivo, id_usuario_instagram)
VALUES ("eTj34sdf8asJH3JHASD...", "cristian123");
```

Respuesta devuelta a Android

Si todo sale bien, el servidor responde:

{
"estado": "ok",
"mensaje": "Usuario registrado correctamente"
}

Si hay error (parámetros vacíos, token inválido, etc.):

{
"estado": "error",
"mensaje": "Faltan parámetros"
}

La app muestra un Toast notificando si el registro se hizo con éxito.

Flujo completo entre Petagram, Firebase y el Backend

A continuación, se explica todo el recorrido de los datos:

Paso 1 – El usuario toca "Recibir Notificaciones"

La aplicación ejecuta:

FirebaseMessaging.getInstance().getToken()

Y obtiene el **id_dispositivo**.

Paso 2 – El token se envía al backend

Retrofit envía una llamada POST a:

https://tu-servidor.herokuapp.com/registrar-usuario

incluyendo el token y el usuario seleccionado.

Paso 3 – El servidor almacena los datos

El backend los recibe y los inserta en la base de datos.

Esto es indispensable para que más adelante podamos enviar notificaciones personalizadas desde Firebase.

Paso 4 – Enviar notificaciones desde Firebase Console

Cuando un usuario ya está registrado en la base, es posible:

- filtrar por token
- enviar mensajes individuales
- enviar mensajes generales

Firebase utiliza ese token para que la notificación llegue **solo a ese dispositivo**.

Conclusiones

- Se implementó un endpoint REST llamado **/registrar-usuario**.
- El endpoint recibe el token FCM y el usuario de Instagram.
- La aplicación Android obtiene ese token desde Firebase Messaging.
- Heroku se utilizó como plataforma de despliegue del servicio backend.
- El servidor almacena la información en la tabla **usuario_instagram**.
- Este proceso permite enviar notificaciones push personalizadas usando Firebase.