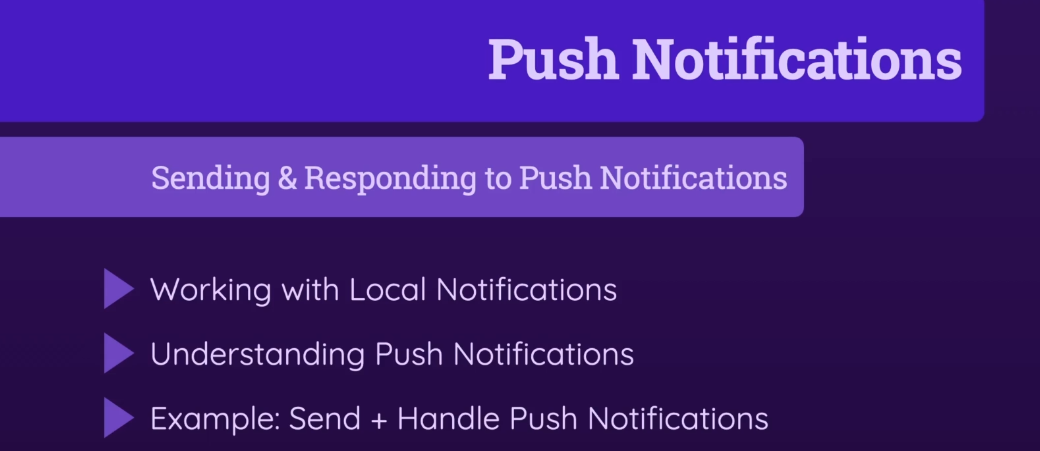
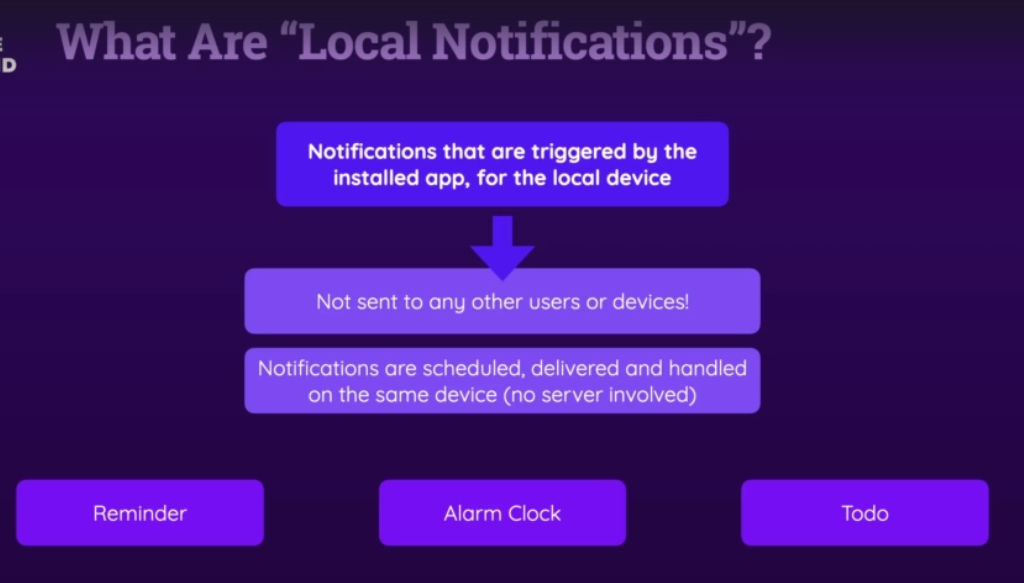
**React Native Push Notifications**



* Notificaciones locales son notificaciones que son activadas por la APP instalada en el mismo dispositivo

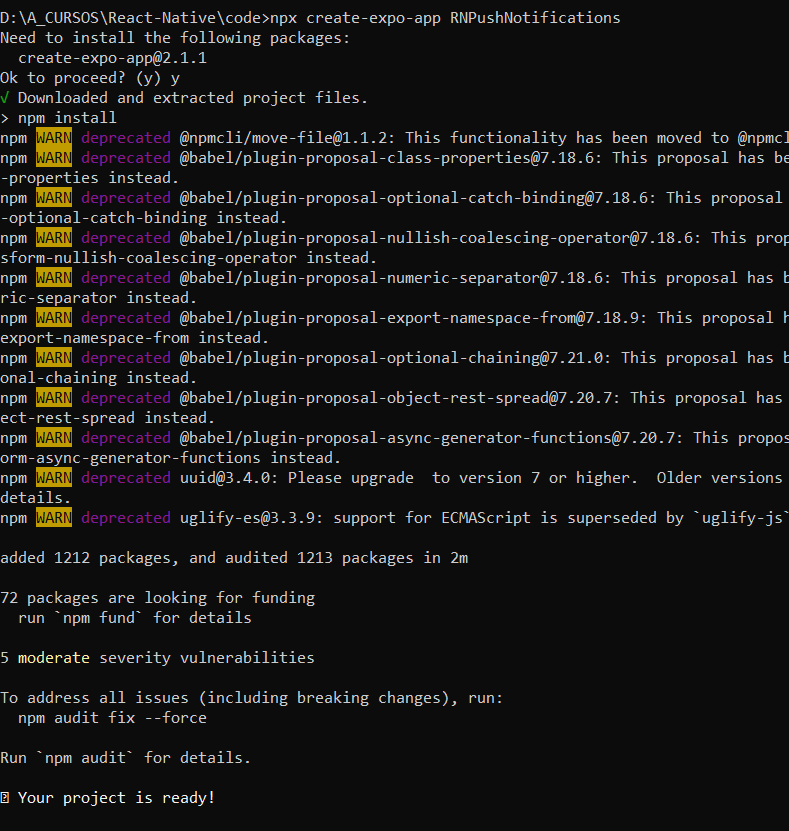


* Solo se puede emplear dentro del mismo dispositivo
* Casos de uso App de recordatorio, de tareas o un despertador o alarmas. Tareas a entregar en el futuro (pendientes)

**Vamos a crear un proyecto base con Expo para configurar luego las push notifications**

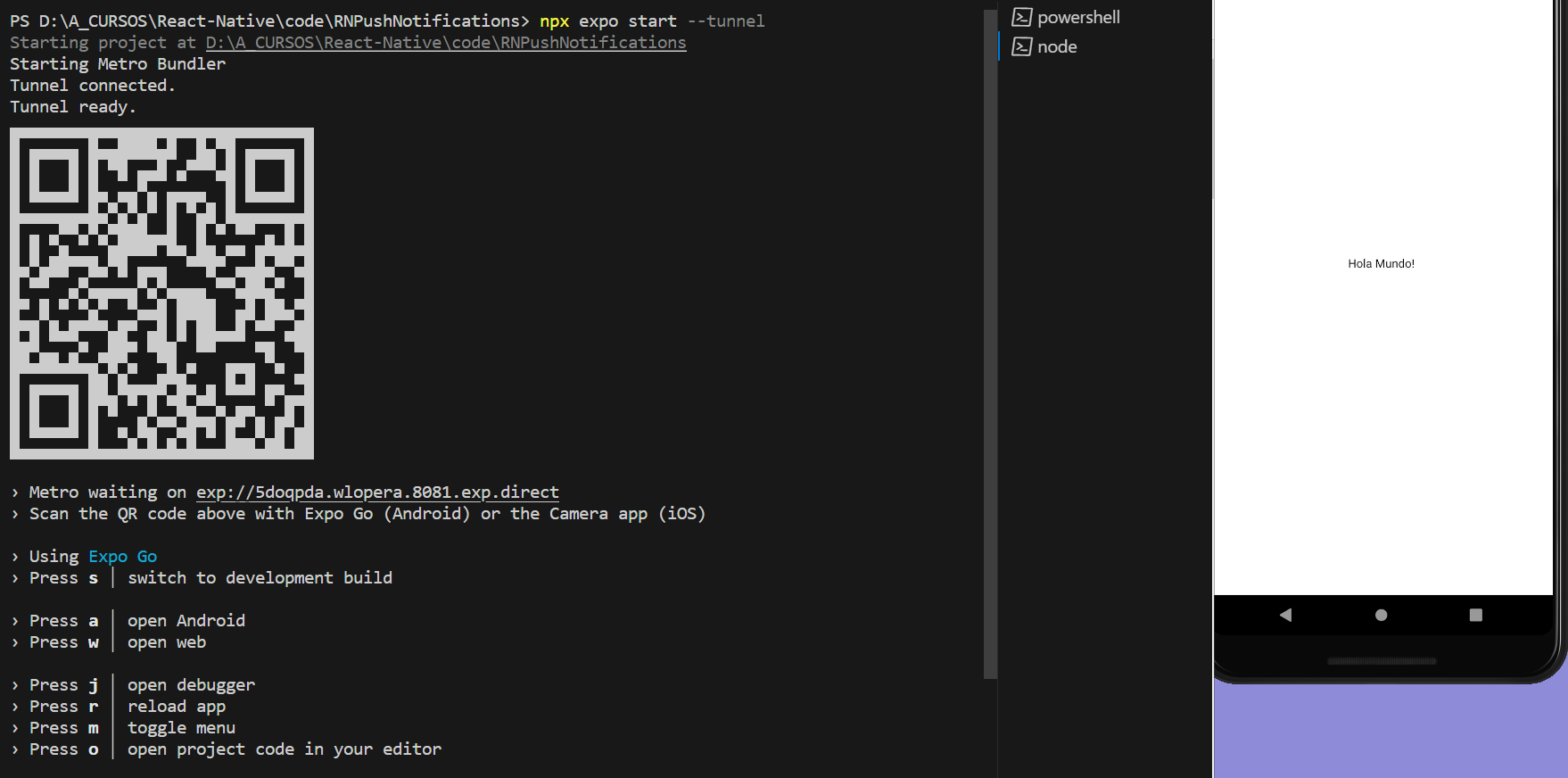
* Creamos un proyecto expo

$> npx create-expo-app RNPushNotifications



* Abrimos el proyectp0 en VSCODE y levantamos el APP. En una terminal del APP (VSCODE)

$> npx expo start –tunnel

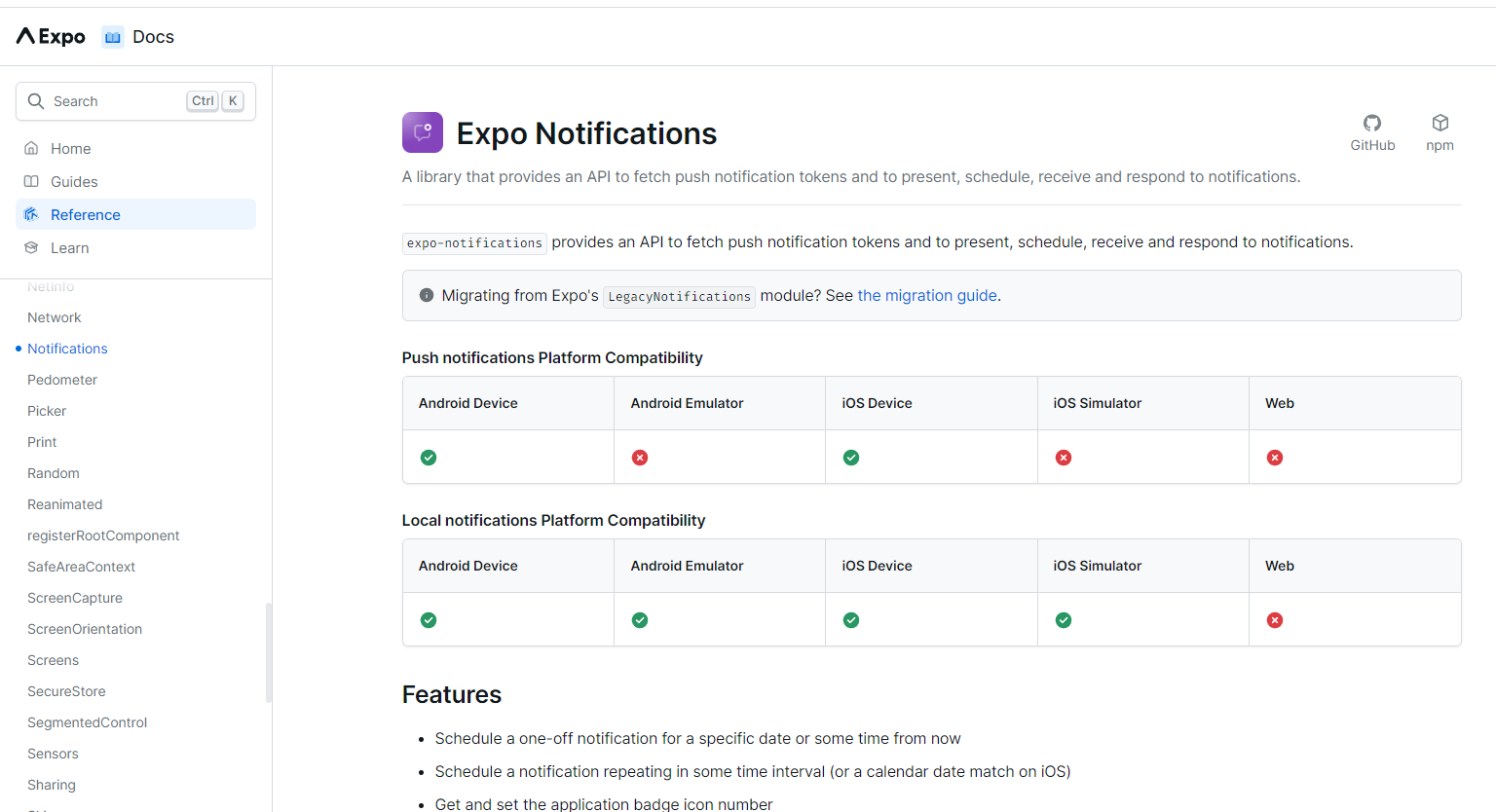


Uso de Simulador de Android Studio



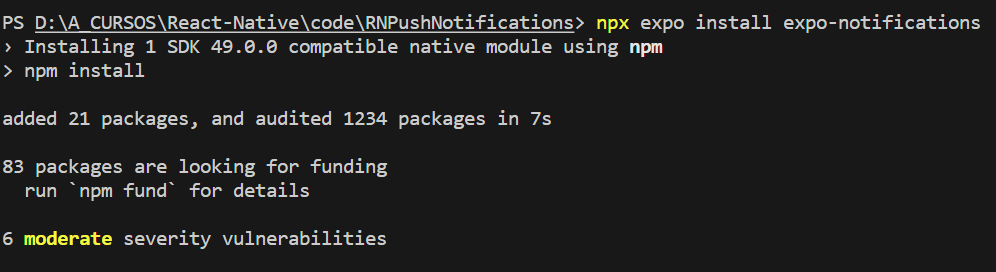
* Uso de Notificaciones de Expo

<https://docs.expo.dev/versions/latest/sdk/notifications/>

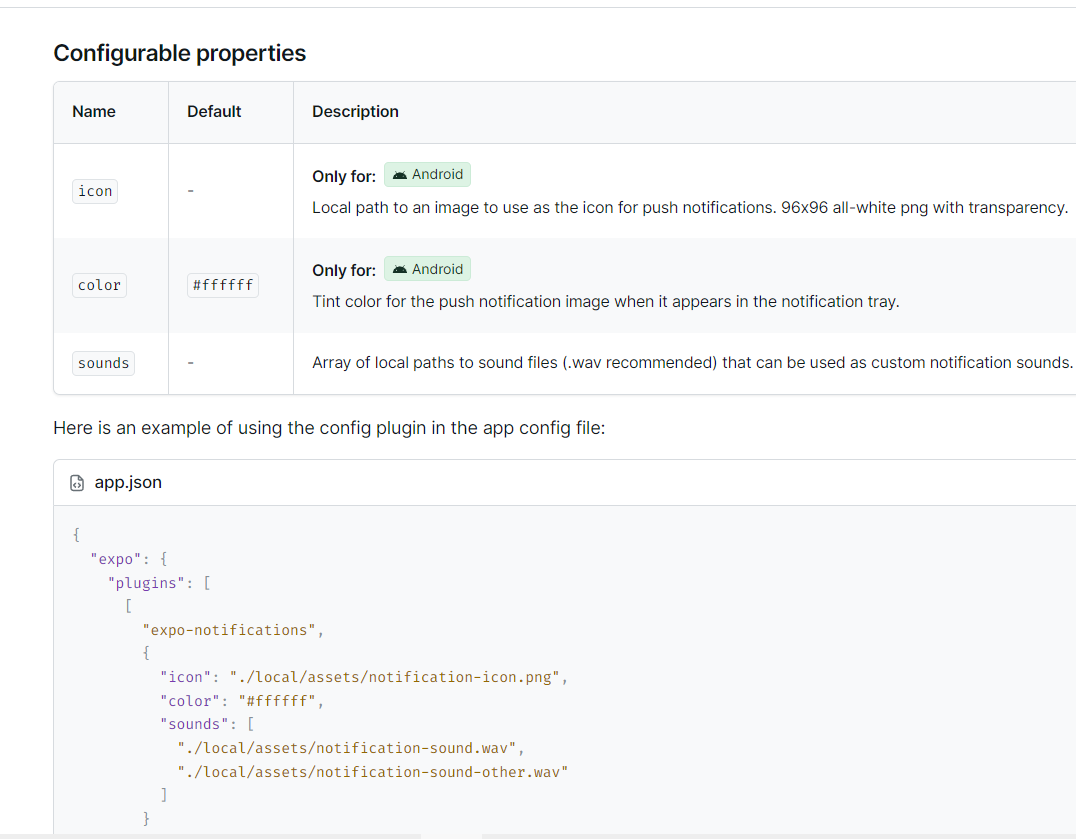


* Instalamos la librería de notificaciones de EXPO

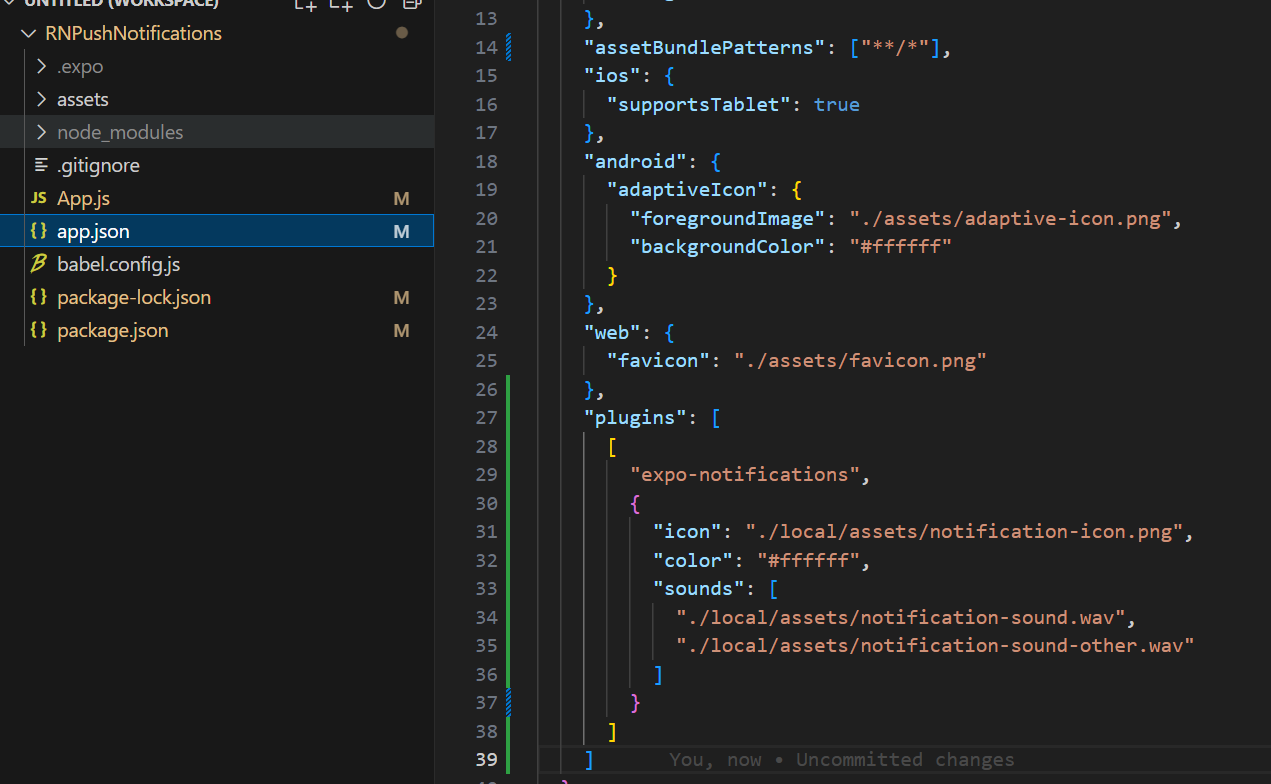
$> npx expo install expo-notifications



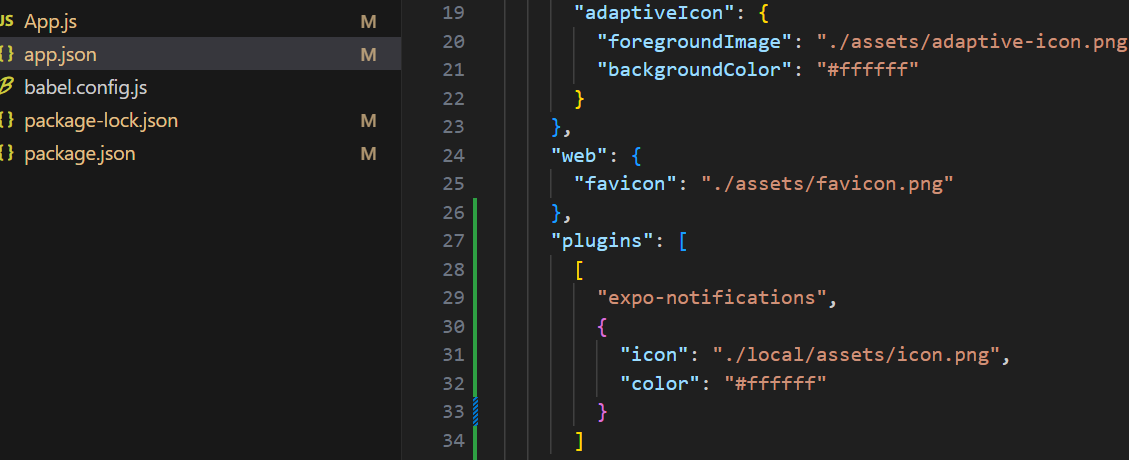
* En la documentación Podemos ver como cambiar el icono y el sonido de las notificaciones



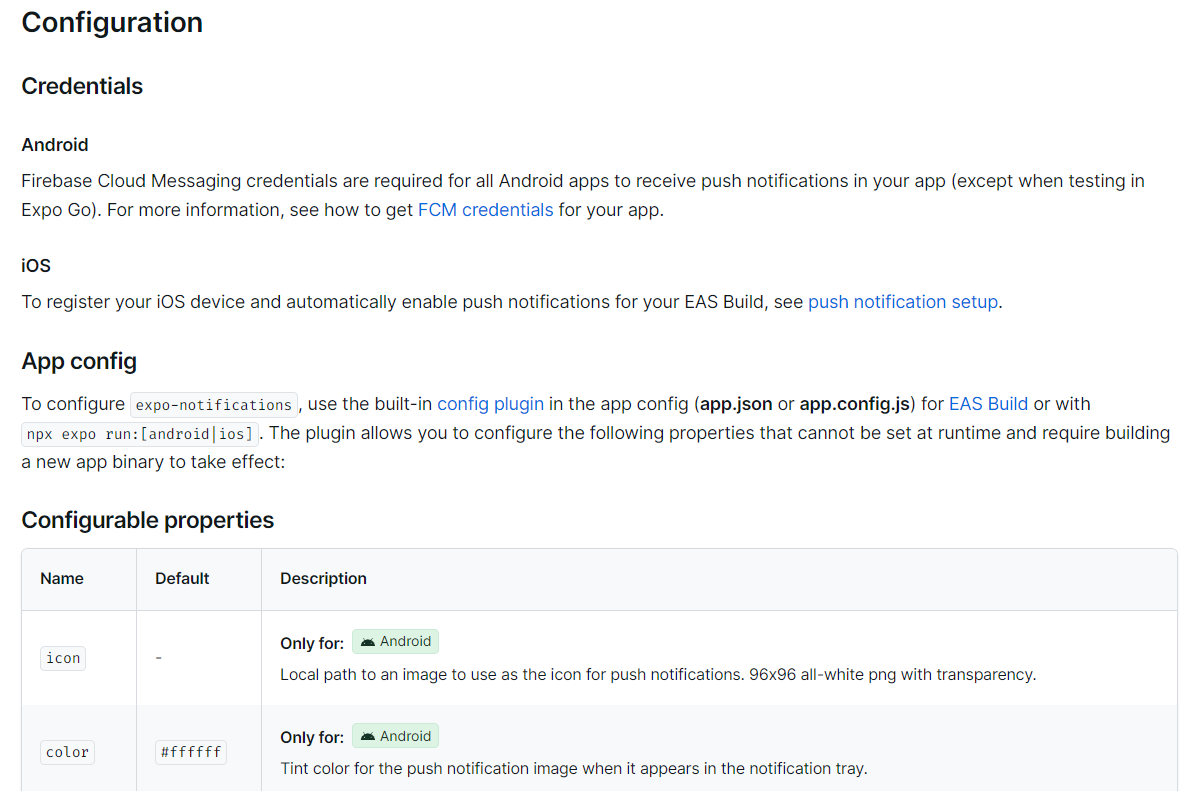
* Se agrega la configuración de icono, color y sonido a nuestro app.json para cambiar al llegar notificaciones



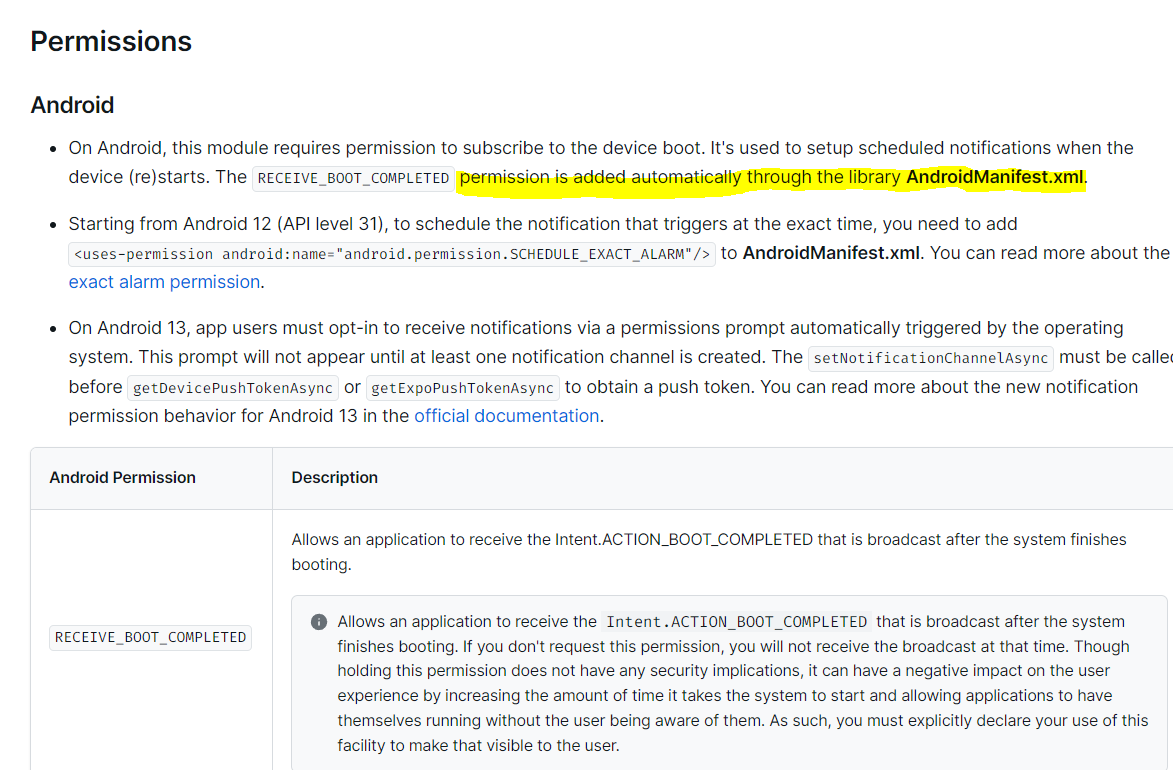
* Voy a dejar solo el icono y el color



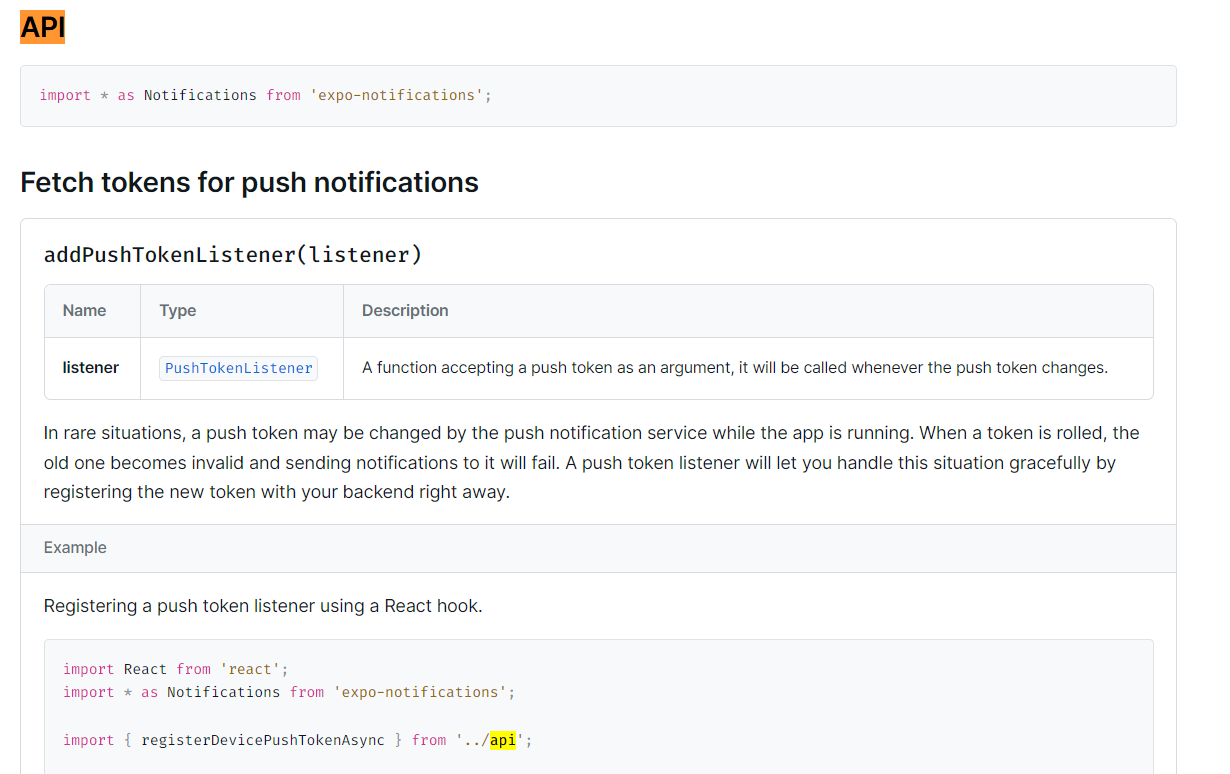
* También existen instalación para configuración extra para APP Expo Básicas o Nativas
* Para App independientes de Expo se deben realizar mas configuraciones y manejo de credenciales

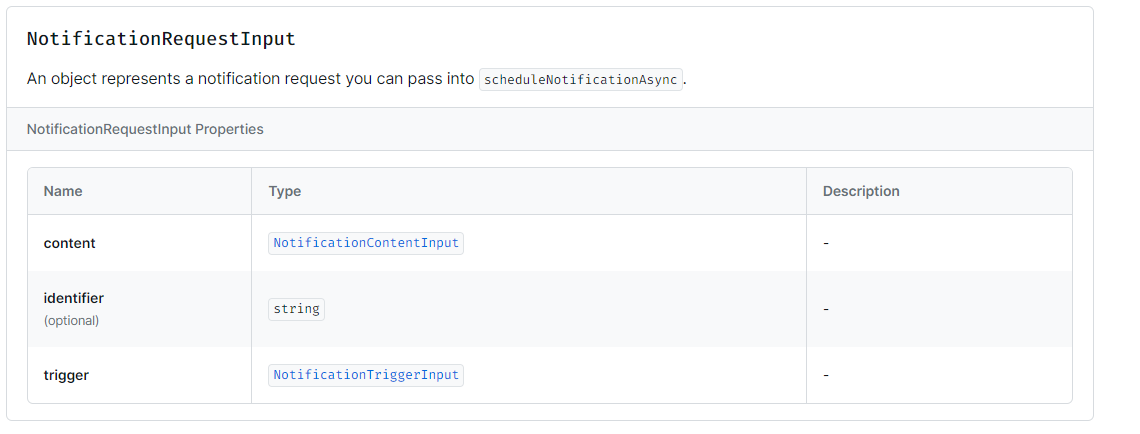


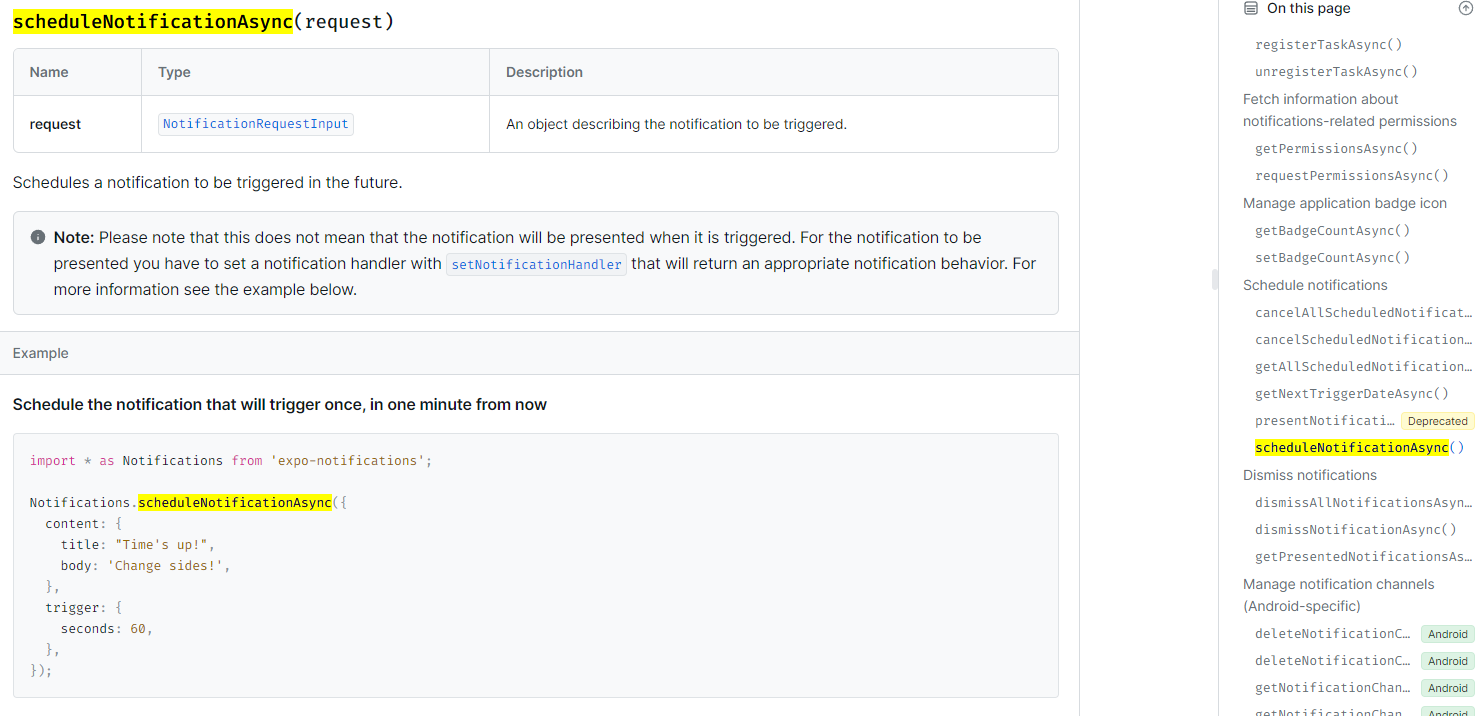
* Los permisos para Android se agregan automáticamente



* Agregar notificaciones. Uso de función de notificaciones de programaciones asíncrona







* Agrego contenido de la notificación. Puedo pasar variables (data: usuario) que puedo consultar luego en el desarrollo

…

export default function App() {

  function scheduleNotificationHandler() {

    Notifications.scheduleNotificationAsync({

      content: {

        title: "NOTIFICACION",

        body: "Cuerpo de la notificación",

        data: { ususario: "wlopera" },

      },

…

* Podría agregar configuración del sonido por código y otras opciones (ver documentación)
* Debemos configurar un disparador del evento notificación. Agregarle un tiempo de 5 seg para lanzar la notificación

...

export default function App() {

  function scheduleNotificationHandler() {

    Notifications.scheduleNotificationAsync({

      content: {

        title: "NOTIFICACION",

        body: "Cuerpo de la notificación",

        data: { ususario: "wlopera" },

      },

      trigger: {

        seconds: 5,

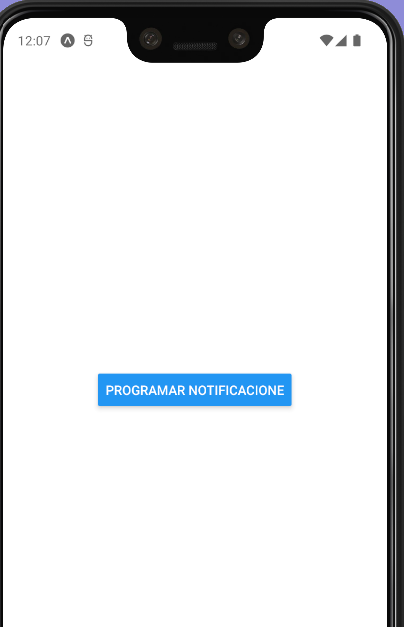
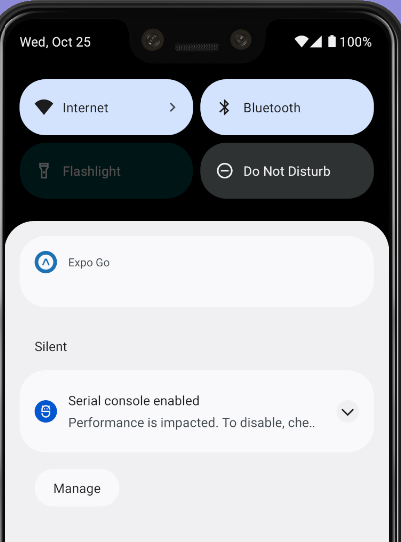
      },

    });

  }

…

* Se levanta la APP y se prueba la notificación se envía a los 5 seg pero no se está recibiendo

Configuring Scheduled Notifications

In the previous lecture, you learned how to schedule (local) notifications via scheduleNotificationAsync().

That method takes one argument - an object where you set the content and the trigger.

Besides the properties shown in the previous lecture, other properties could be added.

You find a complete list (+ explanations) of content properties here: <https://docs.expo.dev/versions/latest/sdk/notifications/#notificationcontentinput>

And a complete list (+ explanations) of trigger properties here: <https://docs.expo.dev/versions/latest/sdk/notifications/#notificationtriggerinput>

The trigger type may be confusing - it's basically a combination of multiple supported object types.

You could either set an interval in seconds (as we did in the previous lecture) as described here: <https://docs.expo.dev/versions/latest/sdk/notifications/#timeintervalnotificationtrigger>

Or you set a specific date (incl. time) at which the notification will be delivered: <https://docs.expo.dev/versions/latest/sdk/notifications/#datetriggerinput>

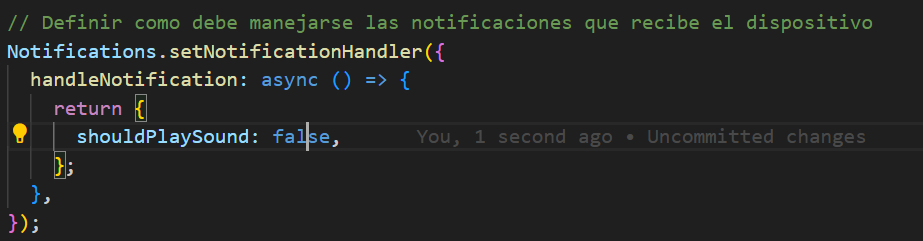
Or you set a daily time at which the notification will be sent (Android-only): <https://docs.expo.dev/versions/latest/sdk/notifications/#dailynotificationtrigger>

Or a weekly trigger (Android-only): <https://docs.expo.dev/versions/latest/sdk/notifications/#weeklynotificationtrigger>

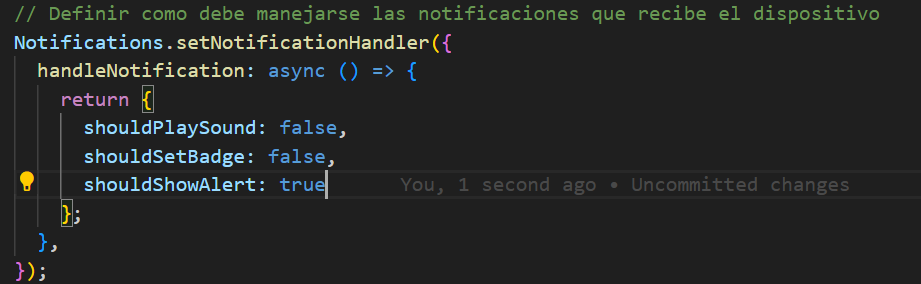
Or a yearly trigger (Android-only): <https://docs.expo.dev/versions/latest/sdk/notifications/#yearlynotificationtrigger>

Or a specific date (iOS-only): <https://docs.expo.dev/versions/latest/sdk/notifications/#calendarnotificationtrigger>

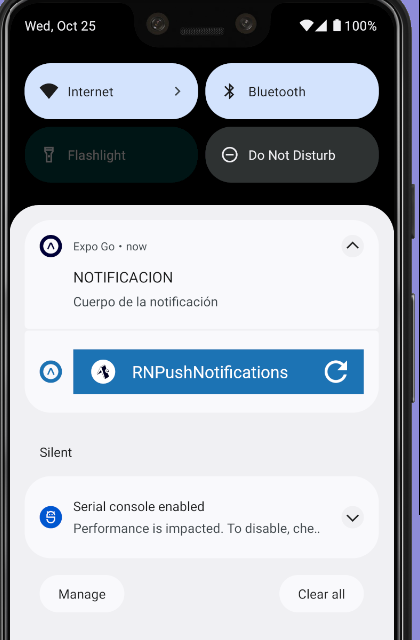
* Se debe agregar una función fuera de la función componente de mi App (para que solo se ejecute cunado se inicie el APP y no cada vez que refresque el APP10/26/2023) que identifique el tipo de notificación y reporte al APP y al dispositivo de la misma como debe manejar las notificaciones entrantes. Por estar la función en App.js solo se corre al iniciar el APP
* La función debe manejar error y éxito porque es el punto de partida de configuración de las notificaciones
* En la función de éxito se puede programar en este caso el sonido. Lo vamos a pasar falso



* Configurar manejo de icono y sonido



* Si recargamos nuestra APP y click en envió de notificación, esperamos 5 seg y debemos ver la notificación en nuestro dispositivo





Local Notifications - Permissions

When using Expo Go, you shouldn't need to ask for any permissions to send or show local notifications (or notifications in general).

This will change as you build your app for production though. Even when using Expo's managed workflow, you will then leave the Expo Go app (as a standalone app will be built by EAS - see section 14).

To ensure that notifications work correctly, you should therefore ask for permission. For Android, no changes are required. For iOS, you can use the getPermissionsAsync() method ([documentation](https://docs.expo.dev/versions/latest/sdk/notifications/#getpermissionsasync-promisenotificationpermissionsstatus)) provided by expo-notifications to get the current permission status. You can use requestPermissionsAsync() ([documentation link](https://docs.expo.dev/versions/latest/sdk/notifications/#requestpermissionsasyncrequest-notificationpermissionsrequest-promisenotificationpermissionsstatus)) to request permissions.

A full code example can be found [here](https://docs.expo.dev/push-notifications/push-notifications-setup/) but **I will also walk you through the entire permission setup later in this section as well.**

* Para controlar los eventos de Request y Response de las notificaciones se utilizan métodos que controlan esos eventos. Para ello se debe utilizar un método (useEffect) que permite detectar las notificaciones y al terminar de usar el componente libere el evento o acción

  …

useEffect(() => {

    const suscription1 = Notifications.addNotificationReceivedListener(

      (notification) => {

        console.log(

          "Notificación peticion recibida:",

          JSON.stringify(notification, null, 2)

        );

        const username = notification.request.content.data.ususario;

        console.log(username);

      }

    );

    const suscription2 = Notifications.addNotificationResponseReceivedListener(

      (response) => {

        console.log(

          "Notificación respuesta recibida:",

          JSON.stringify(response, null, 2)

        );

        const username = response.notification.request.content.data.ususario;

        console.log(username);

      }

    );

    return () => {

      suscription1.remove();

      suscription2.remove();

    };

  }, []);

…

* Cuando se recibe la notificación (Request)

Notificación peticion recibida: {

"request": {

"trigger": {

"seconds": 5,

"channelId": null,

"repeats": false,

"type": "timeInterval"

},

"content": {

"title": "NOTIFICACION",

"badge": null,

"autoDismiss": true,

"data": {

"ususario": "wlopera"

},

"body": "Cuerpo de la notificación",

"sound": "default",

"sticky": false,

"subtitle": null

},

"identifier": "55263ceb-185d-4438-b79e-7b04b4347aff"

},

"date": 1698245665971

}

LOG wlopera

* Cuando se recibe la respuesta de la notificación (Response). Al darle click a la notificación

LOG Notificación respuesta recibida: {

"notification": {

"request": {

"trigger": {

"seconds": 5,

"channelId": null,

"repeats": false,

"type": "timeInterval"

},

"content": {

"title": "NOTIFICACION",

"badge": null,

"autoDismiss": true,

"data": {

"ususario": "wlopera"

},

"body": "Cuerpo de la notificación",

"sound": "default",

"sticky": false,

"subtitle": null

},

"identifier": "55263ceb-185d-4438-b79e-7b04b4347aff"

},

"date": 1698245665971

},

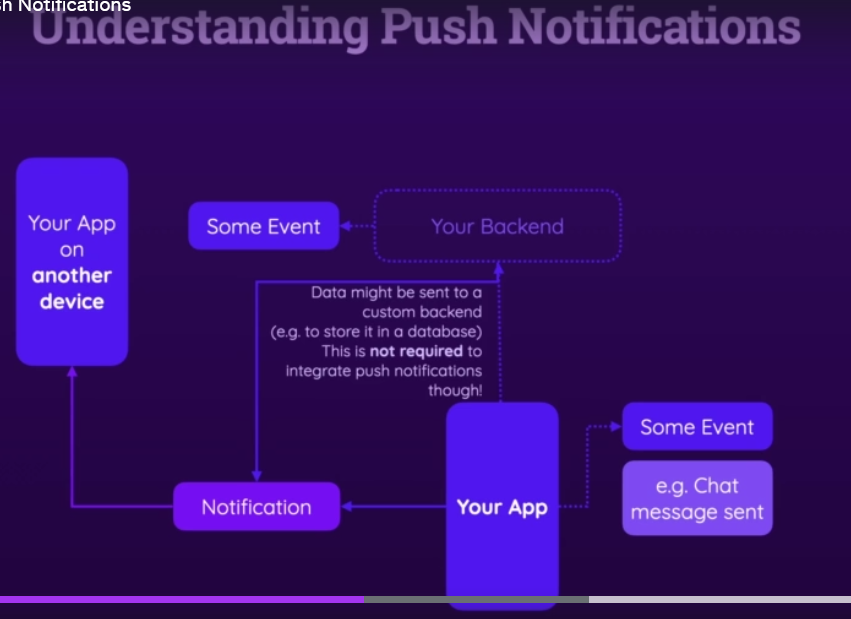
"actionIdentifier": "expo.modules.notifications.actions.DEFAULT"

}

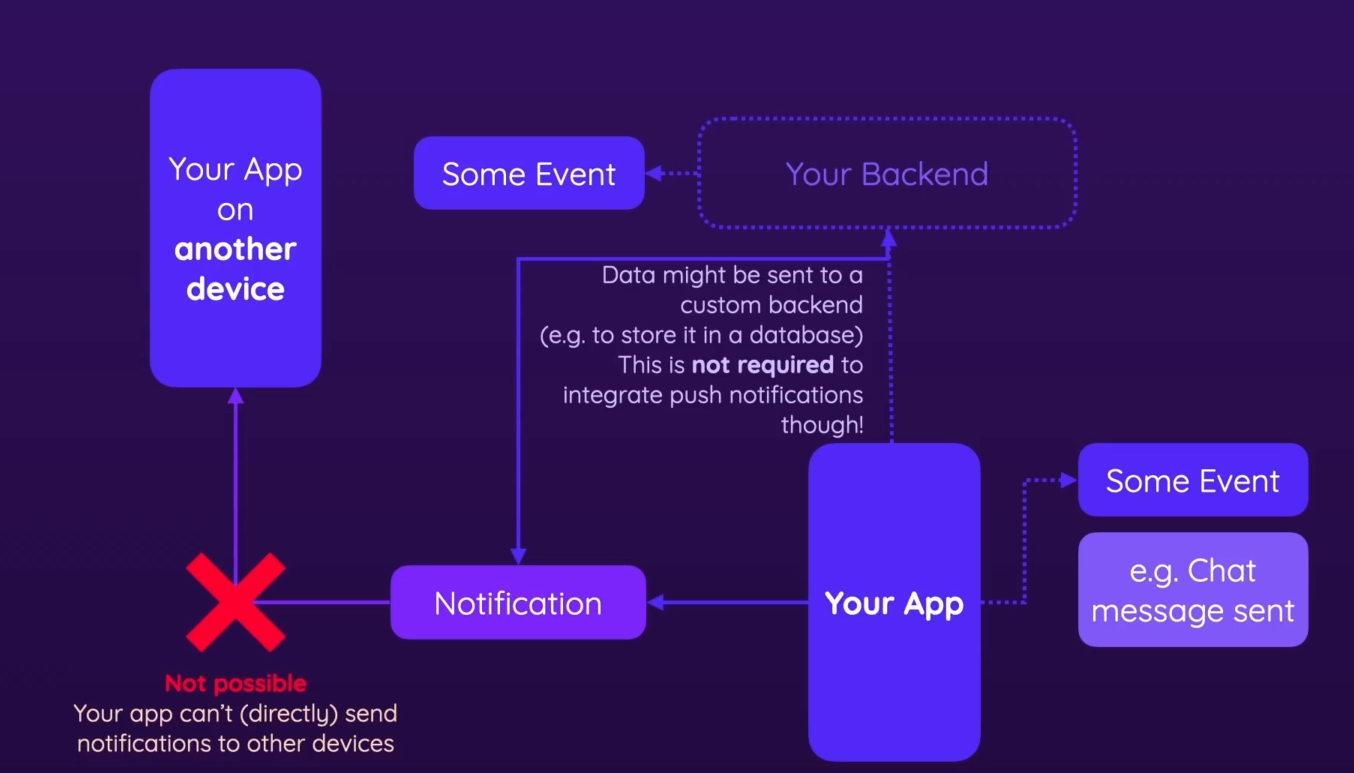
LOG wlopera

**Push Notifications (Notificaciones automáticas)**

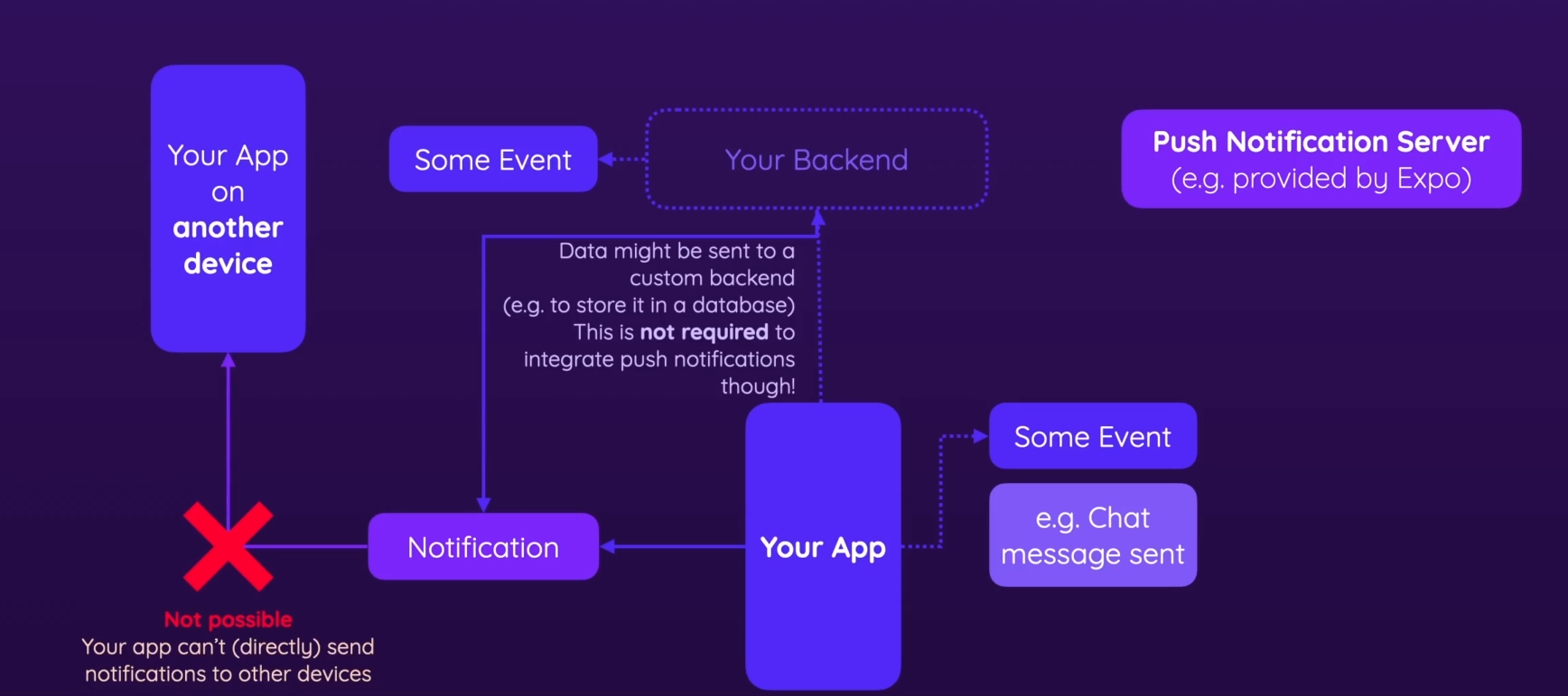
* Puede que nuestro Backend o desde nuestro dispositivo o de algún otro medio externo se envié una notificación (ejemplo: chat o información de marketing) a nuestro dispositivo.



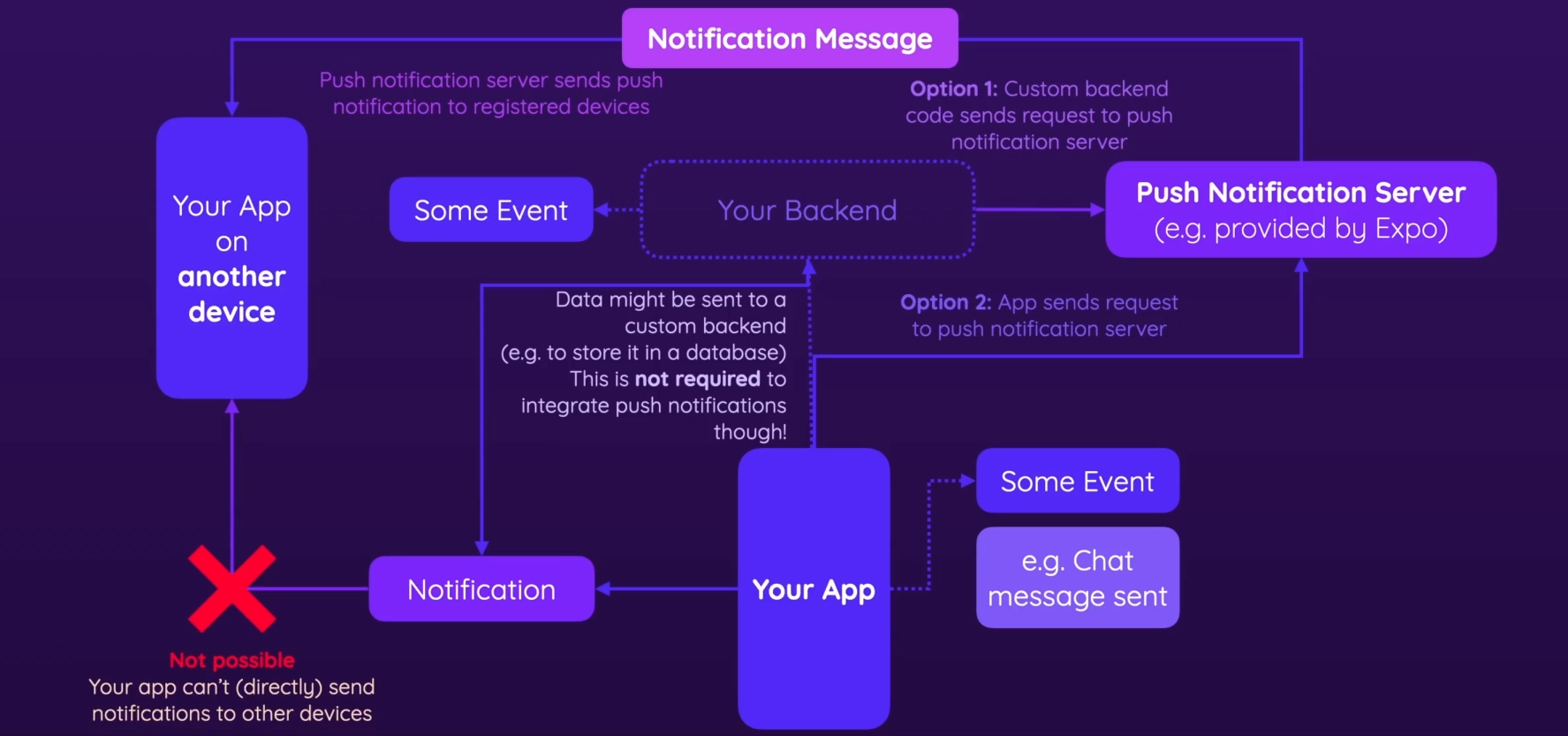
* Se puede enviar mensajes a un dispositivo o a cientos o miles de dispositivos
* Por medida de seguridad, un dispositivo no puede enviar notificaciones directamente a otro dispositivo



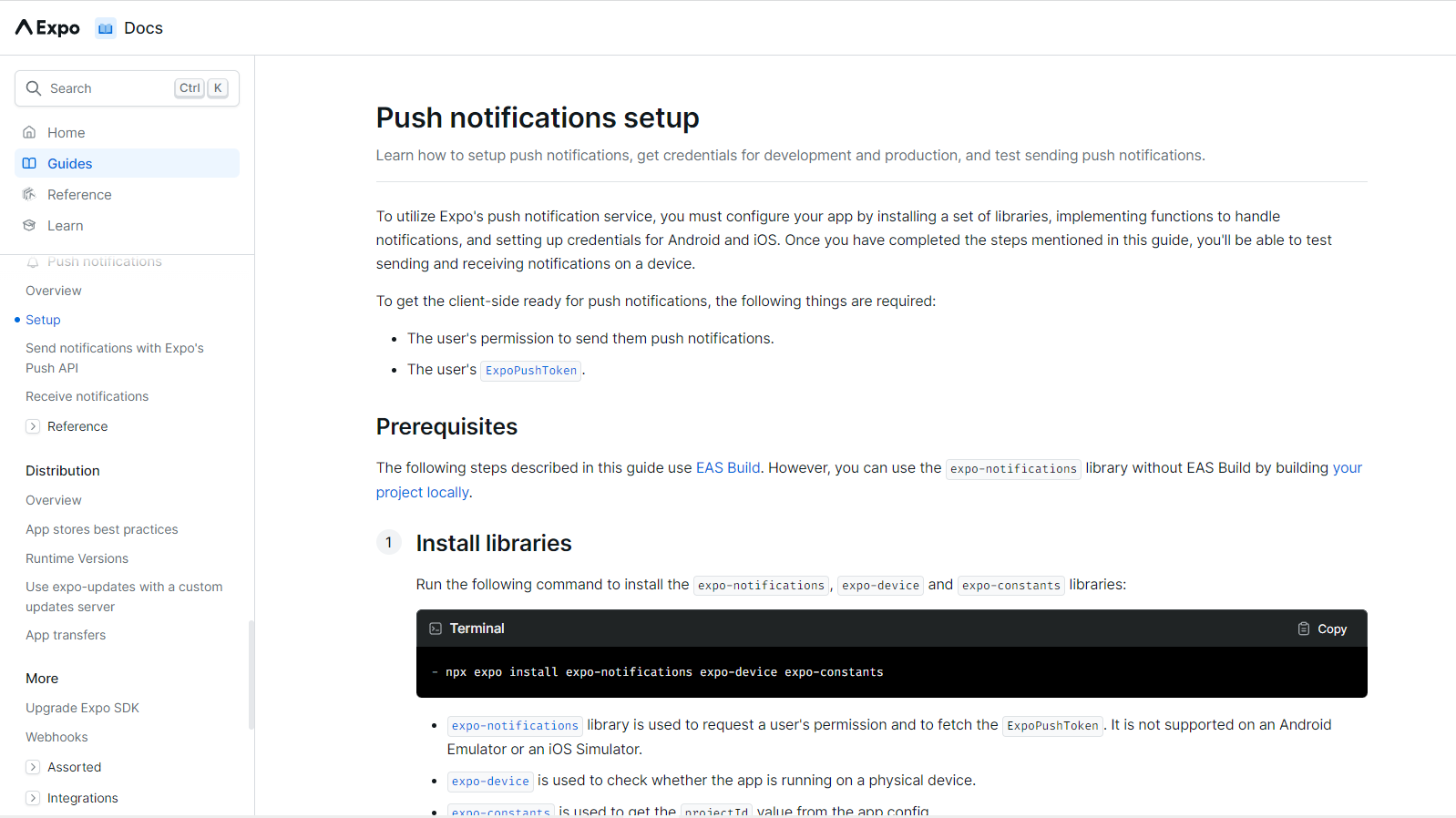
* Uso de proveedores para enviar Push Notifications, por ejemplo, EXPO (Expose Push Notification Server) permite hablar con un servidor para enviar notificaciones a múltiples dispositivos y múltiples plataformas



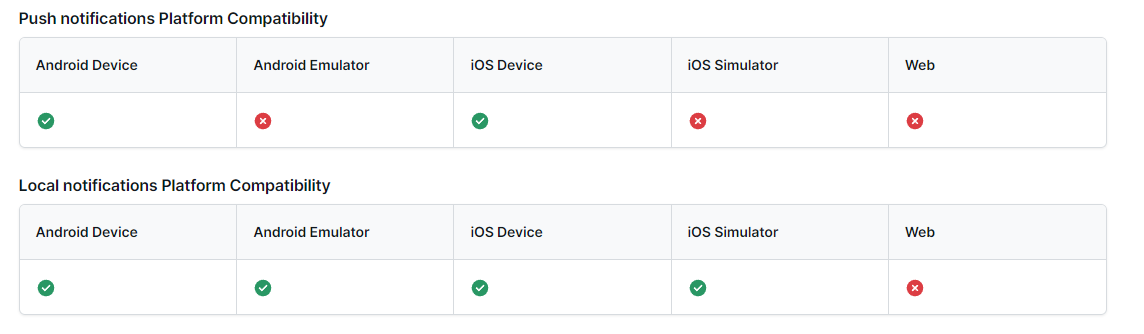
* El Backend puede solicitar a Expo el envió de notificaciones automáticas o se puede hacer directamente desde el dispositivo. El Servidor de Expo a través de Google o Apple quienes son los que se comunican con los dispositivos y envían las notificaciones



* Se debe utilizar la documentación. Para poder enviar notificaciones automáticas desde Backend o a través del dispositivo uso del Servidor de Expo

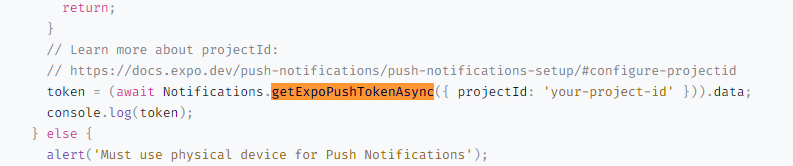


* Se debe generar un Token para enviar el Push Notifications
* Para las notificaciones automáticas debemos utilizar dispositivos reales
* Cada dispositivo IOS o Android debe generar su propio token
* La idea es identificar los dispositivos a los que se le debe enviar los mensajes. Y la dirección de cada dispositivo sería el token de inserción de cada dispositivo
* Recuerda que Push Notifications solo se puede correr en dispositivos reales

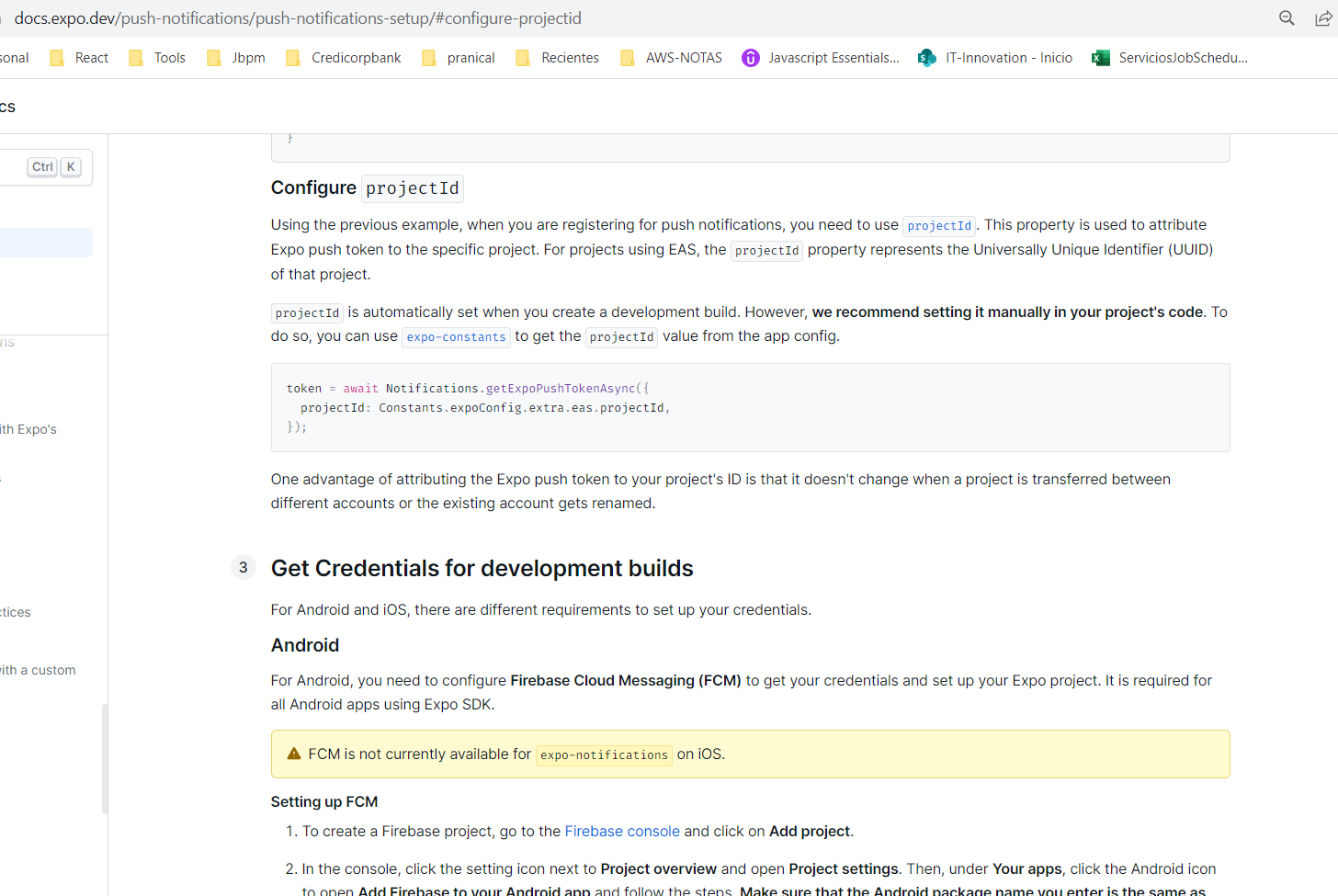


* Se debe solicitar los permisos (requeridos por IOS) y en el caso de Android la función

getExpoPushTokenAsync, requiere un projectId



<https://docs.expo.dev/push-notifications/push-notifications-setup/#configure-projectid>

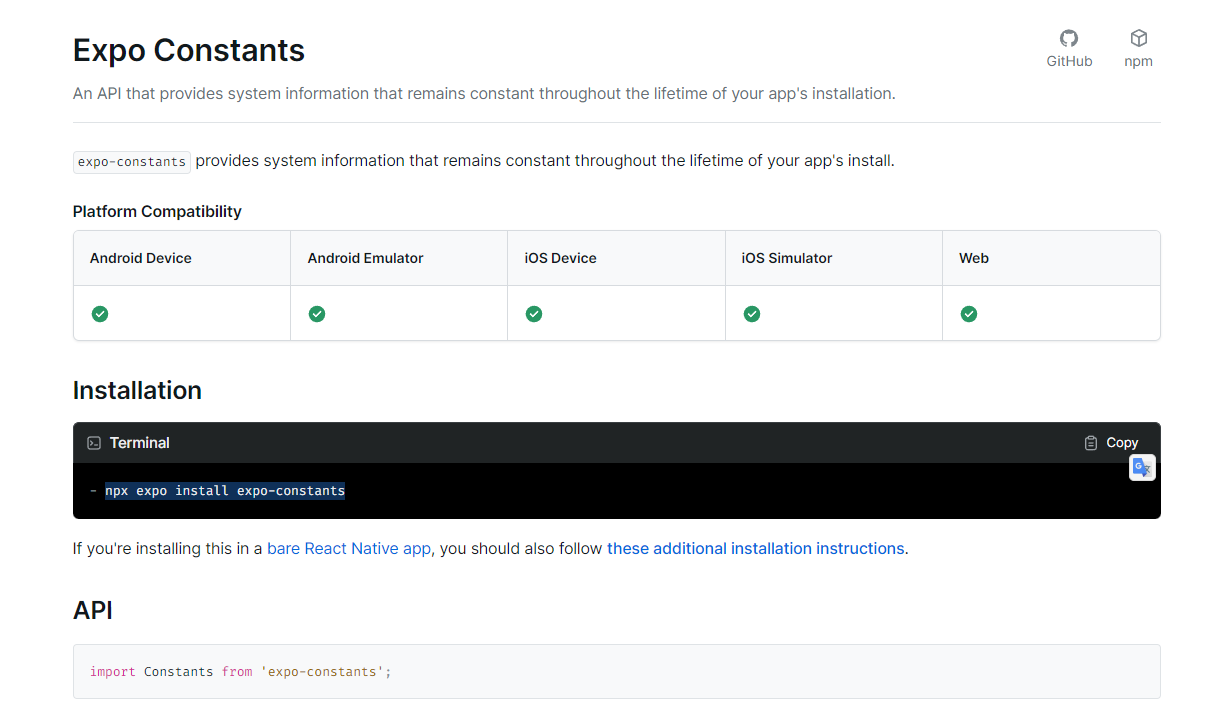


* Se debe obtener el projectId de expo-constants de la configuración de la APP

token = await Notifications.getExpoPushTokenAsync({

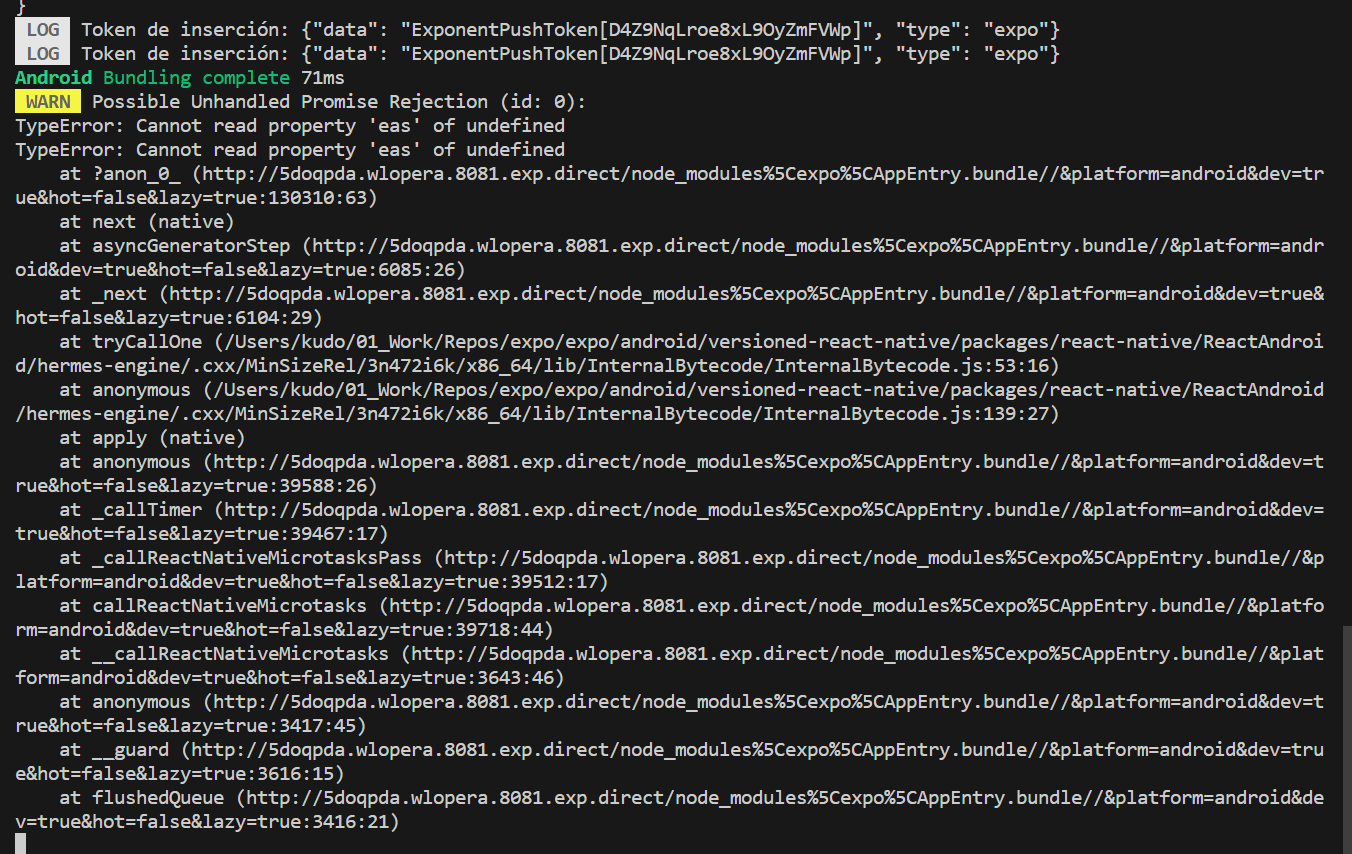
projectId: Constants.expoConfig.extra.eas.projectId,

});

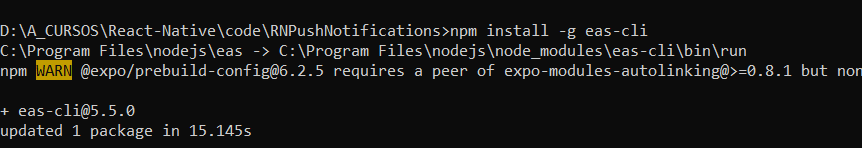


$> npx expo install expo-constants

* Se levanta el App pero al subir al dispositivo da error



* Se debe crear un projectid
  + Installar eas-cli

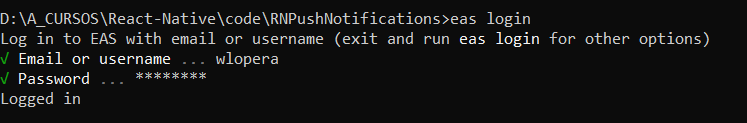


* Conectarse a EAS

$> eas Login

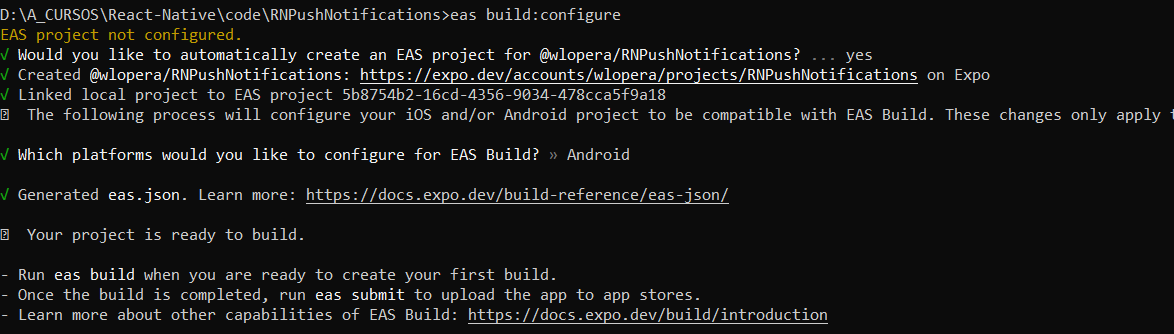
Username: wlopera

Password: Expo\_123

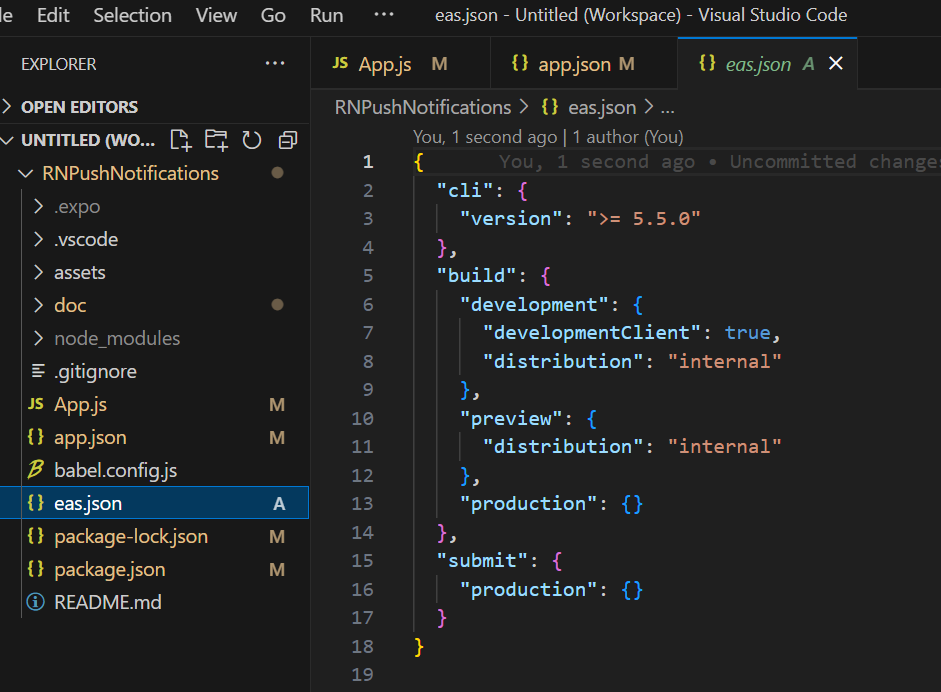


* Configurar el Build

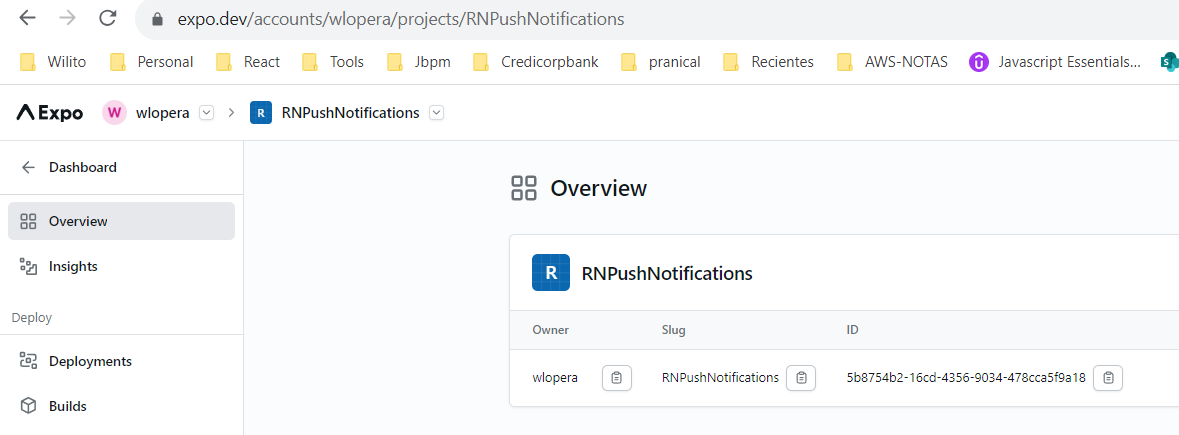
$> eas build:configure



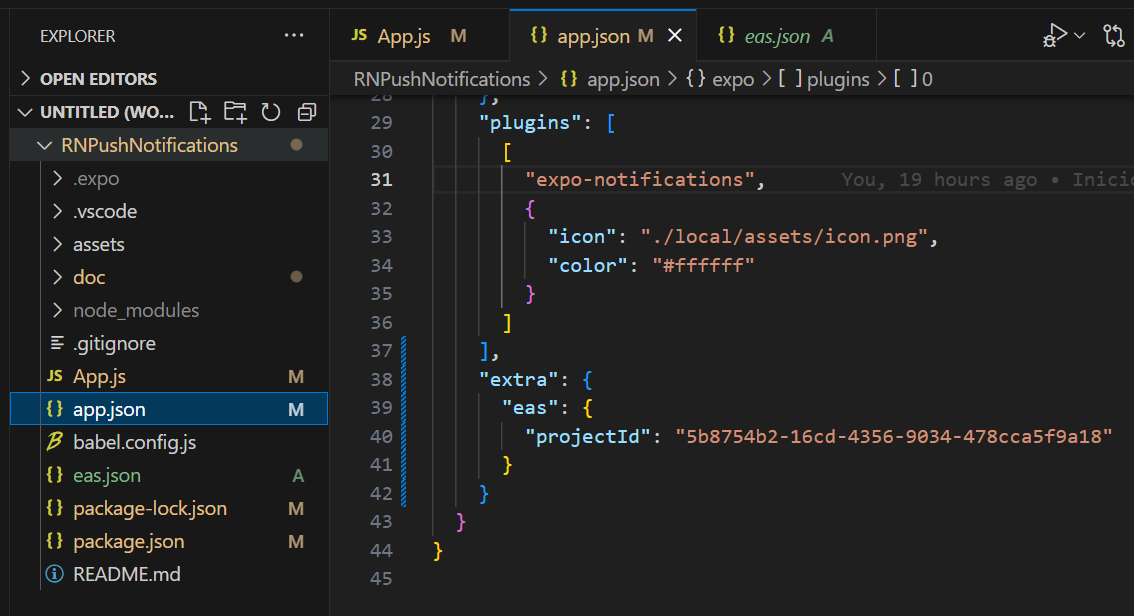
* Se genera el eas.json



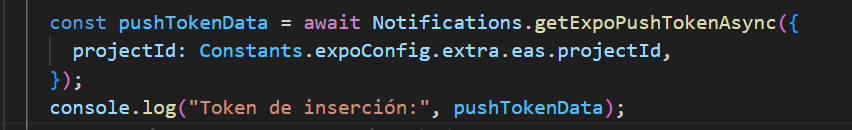
* Se agrega el proyecto a EAS

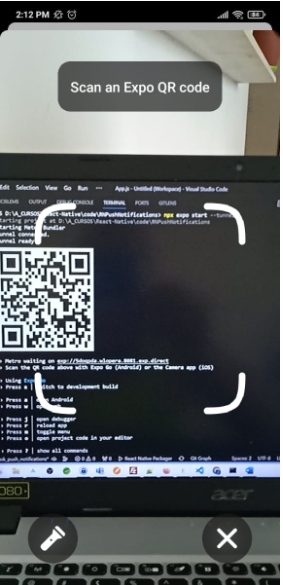


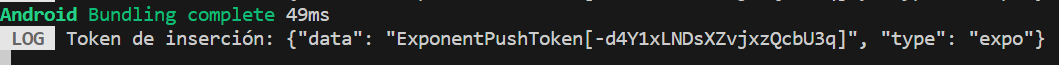
* Se agrega el projectid al app.json



* Ahora si levanto el APP se despliega el Token del APP en mi Dispositivo Real: Redmi 9A

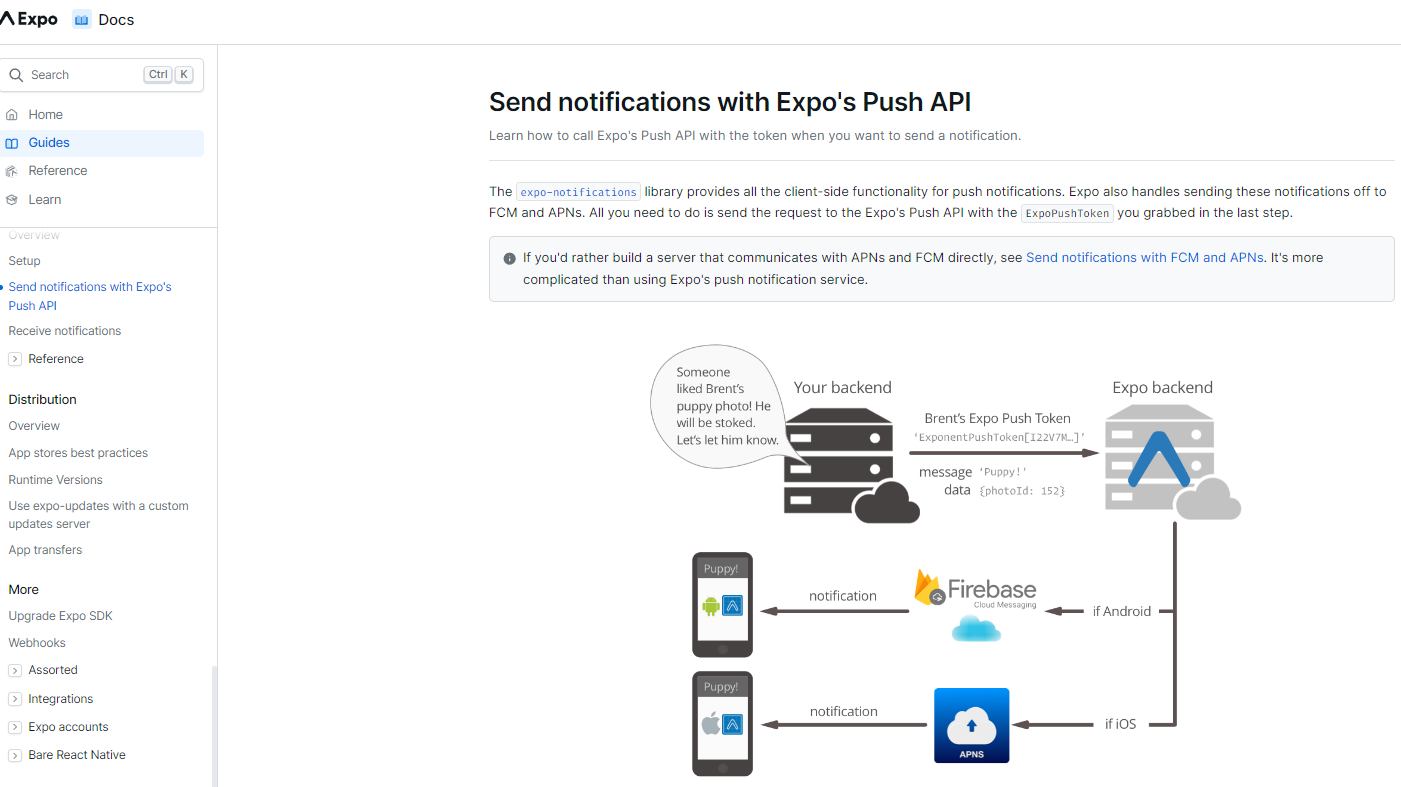






Token de inserción: {"data": "ExponentPushToken[-d4Y1xLNDsXZvjxzQcbU3q]", "type": "expo"}

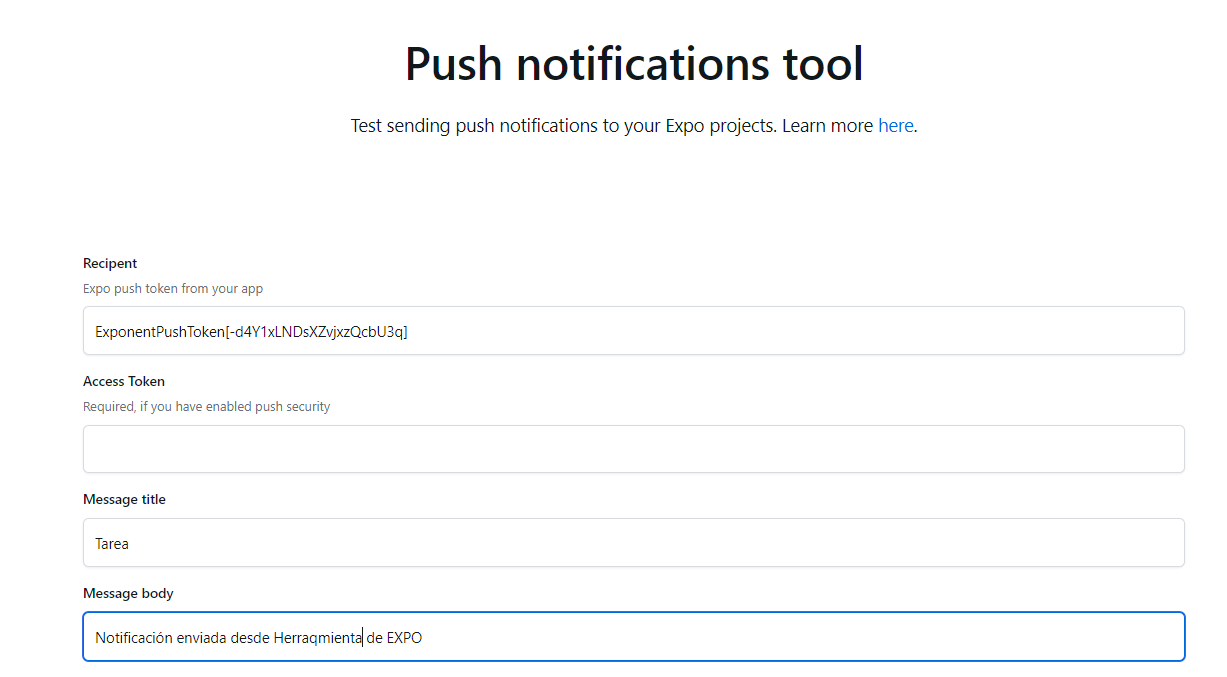
* Enviar notificaciones

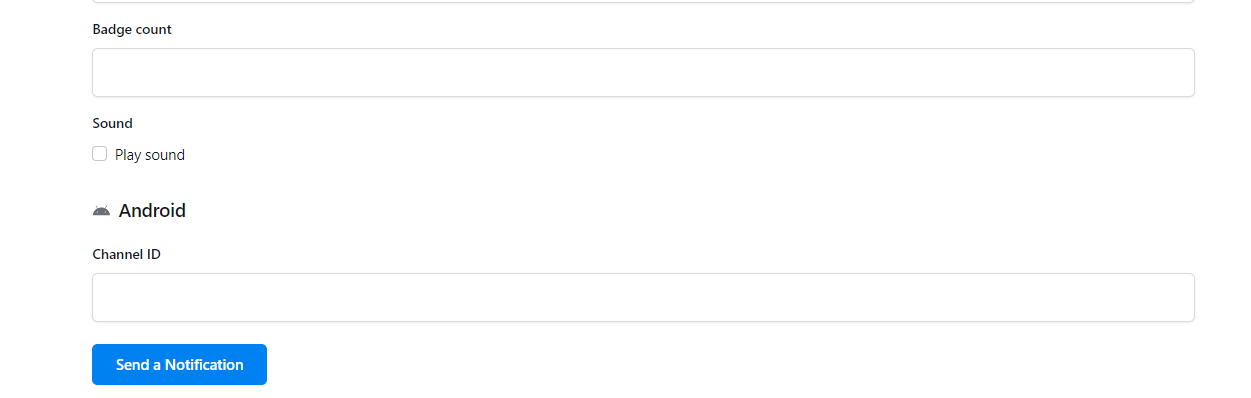


<https://expo.dev/notifications>

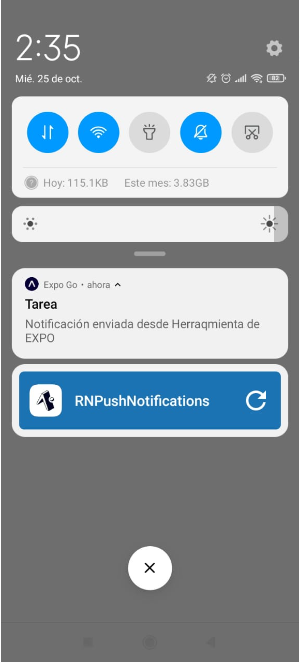


* Utilizar herramienta de Expo Notifications para enviar notificaciones
* Tomar token enviado por Expo ExponentPushToken[-d4Y1xLNDsXZvjxzQcbU3q]

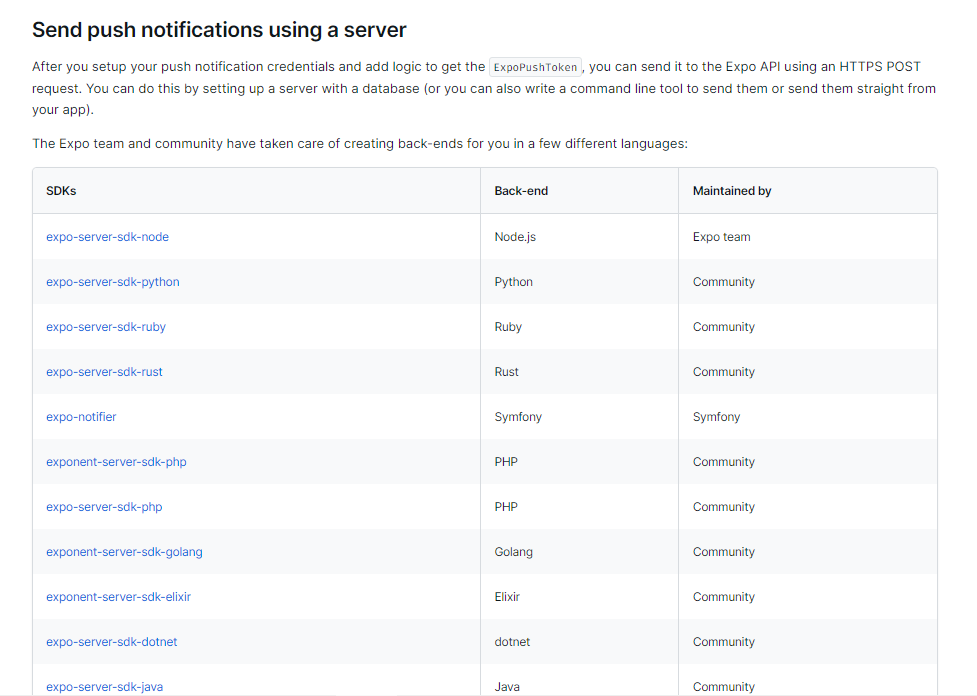


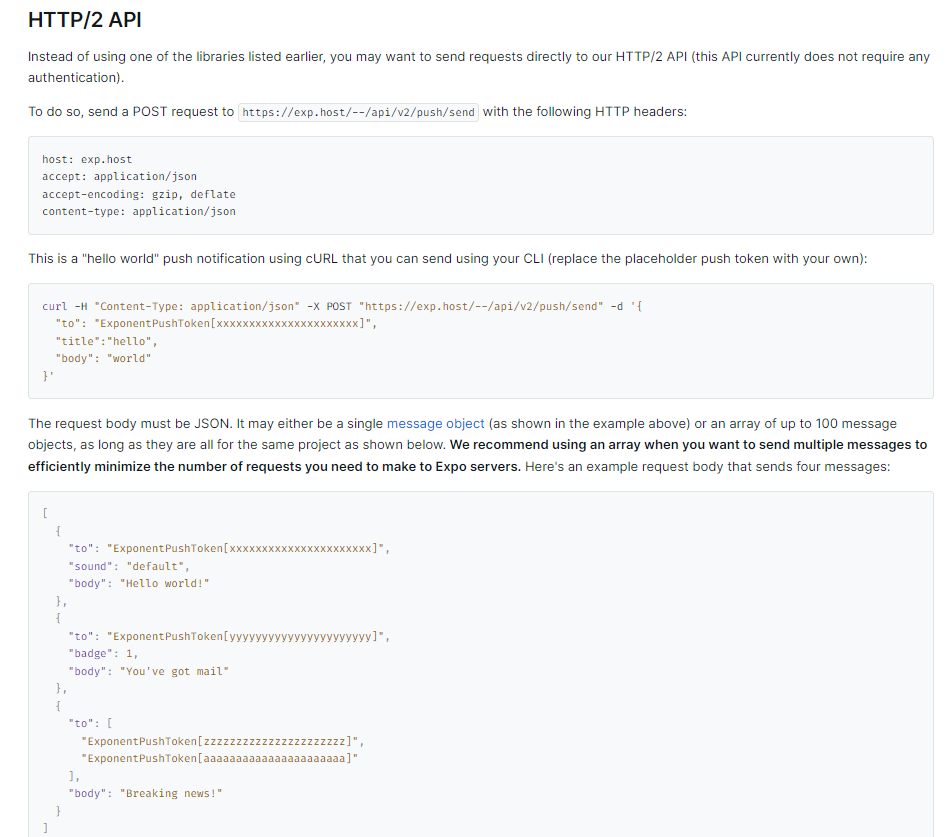


* Se envía y llega el Push Notification a mi dispositivo real



* Para enviar Push Notifications por desarrollo (Backend) ver documentación



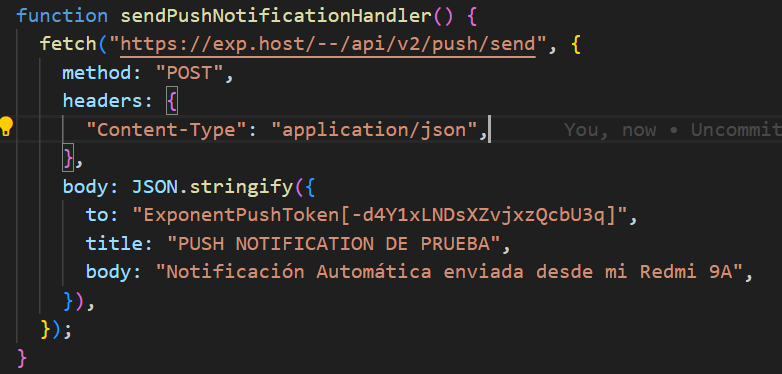




* También podemos hacerlo desde nuestra interfaz
* Vamos a consumir un servicio API Send Notification de Expo desde fetch

To do so, send a POST request to https://exp.host/--/api/v2/push/send with the following HTTP headers:

* Los tokens que se generen se pueden almacenar en una base de datos y luego consultarlos para enviar notificaciones programadas desde el dispositivo
* A modo de demostración se va a enviar una Push Notification desde el dispositivo colocando el Token directamente, pero este valor debería venir de una DB o servicio que consulte la información (Token) almacenada.



* Subo y pruebo el APP  
   