**1. Use OneSignal for Push Notification instead of Firebase**

1. Sign into OneSignal dashboard → **New App/Website**.  
   https://dashboard.onesignal.com/login
2. Give it a name and add platforms (Android, iOS). Follow the prompts for each platform. (OneSignal UI walks you through this.)

# **2. Install SDK (React Native)**

From your RN project root:

yarn add react-native-onesignal

# or

npm install react-native-onesignal  
  
**Remove existing firebase packages**   
npm uninstall @react-native-firebase/app

npm uninstall @react-native-firebase/messaging

# or  
yarn remove @react-native-firebase/app

yarn remove @react-native-firebase/messaging

**3. Android setup (FCM credentials & app config)**

### **A. Create Firebase service account JSON (required by OneSignal)**

1. In Firebase Console → Project Settings → Service accounts → Generate new private key → download JSON.
2. In OneSignal dashboard: **Settings → Push & In-App → Push Platforms → Google Android (FCM)** → Upload the **Service Account JSON** you downloaded. OneSignal will use that to send via FCM. **Important:** use the JSON from the Firebase project that owns your current Sender ID (mixing sender IDs breaks subscriptions).

### **B. AndroidManifest & build.gradle checks**

* Ensure android/app/src/main/AndroidManifest.xml includes Internet permission (usually present).
* In android/build.gradle & app/build.gradle ensure no conflicting firebase plugins; OneSignal will integrate with the native SDK. (Follow OneSignal Android SDK doc if you need manual steps.) [OneSignal Documentation](https://documentation.onesignal.com/docs/android-sdk-setup?utm_source=chatgpt.com)

### **C. Add OneSignal App ID**

No JS code yet — add OneSignal App ID in the initialization step below.

# **4. iOS setup (APNs)**

You must upload APNs credentials to OneSignal so OneSignal can deliver pushes to APNs:

Options:

* **APNs Auth Key (.p8)** — recommended (Apple Developer → Keys → create APNs key, download .p8). Upload to OneSignal and provide Key ID & Team ID.
* Or **APNs certificate (.p12)** — older method. [OneSignal Documentation](https://documentation.onesignal.com/docs/ios-sdk-setup?utm_source=chatgpt.com)

Xcode:

* Enable **Push Notifications** for your target.
* Add OneSignal initialization in AppDelegate if manual native setup is required (OneSignal docs show Objective-C/Swift instructions).

# **5. Initialize OneSignal in React Native (JS)**

Create a small module (e.g., onesignal.ts) and init in your app entry (e.g., App.tsx):

// onesignal.ts

import OneSignal from 'react-native-onesignal';

const ONESIGNAL\_APP\_ID = 'YOUR\_ONESIGNAL\_APP\_ID';

export function initOneSignal() {

  OneSignal.setAppId(ONESIGNAL\_APP\_ID);

  // Optional: prompt for push permissions (iOS)

  OneSignal.promptForPushNotificationsWithUserResponse();

  // Handlers

  OneSignal.setNotificationOpenedHandler(notification => {

    console.log('Notification opened:', notification);

    // navigate or handle payload

  });

  OneSignal.setNotificationWillShowInForegroundHandler(notificationReceivedEvent => {

    let notif = notificationReceivedEvent.getNotification();

    console.log('Foreground notification:', notif);

    // decide whether to show it

    notificationReceivedEvent.complete(notif); // show

  });

  OneSignal.setExternalUserId && OneSignal.setExternalUserId('user-id-if-any');

}

Call initOneSignal() early in your app (e.g., inside root component useEffect). [OneSignal Documentation](https://documentation.onesignal.com/docs/react-native-sdk-setup?utm_source=chatgpt.com)**Remove existing firebase code:**

//remove this line form **initializeApp** function  
Path: **project\_root/src/screens/splashScreen/SplashScreen**  
  
await requestNotificationPermission()    
  
**Remove these two functions:**  
const requestNotificationPermission = async () => {  
try {  
const authStatus = await messaging().requestPermission()  
await notifee.requestPermission()

const enabled = authStatus === messaging.AuthorizationStatus.AUTHORIZED ||authStatus === messaging.AuthorizationStatus.PROVISIONAL

 if (enabled) {await getDeviceToken()}  
} catch (error) {  
console.error('Error requesting notification permission:', error)  
}  
}  
// Function to get device token for FCM  
const getDeviceToken = async () => {  
try {const token = await messaging().getToken()

if (token) {

  await AsyncStorageService.setItem('@fcmToken', token)  
}  
checkLock()  
} catch (error) {  
          console.error('Failed to get device token:', error)  
      }  
}

# **6. Request permission & get device id/token**

You can obtain OneSignal player ID (device id) for server-side targeting:

const deviceState = await OneSignal.getDeviceState();

console.log('playerId (OneSignal):', deviceState?.userId);

console.log('pushToken:', deviceState?.pushToken); // native token (APNs/FCM)

Use deviceState.userId to save device mapping in your backend if required.

# **7. Send test notification**

* Use OneSignal Dashboard → Messages → New Push → select your app → Target → Test Users / All Users → Send.
* Or call OneSignal REST API from the server (requires API key). See OneSignal docs to POST JSON to https://onesignal.com/api/v1/notifications.

For more query refer to oneSignal official documentation: [https://documentation.onesignal.com/docs/react-native-sdk-setup#step-by-step-instructions-for-configuring-your-onesignal-app](https://documentation.onesignal.com/docs/react-native-sdk-setup" \l "step-by-step-instructions-for-configuring-your-onesignal-app)