Publish a React Native application to Expo

There are several ways to deploy a React Native application depending on the target platform (iOS or Android) and the desired deployment method (e.g. App Store, Google Play, or direct installation).

iOS:

Build the iOS app using the react-native run-ios command in the project directory.

1. Open the Xcode project located in the ios folder of your project.

2. In Xcode, select the app's target and then select the "General" tab.

3. Under the "Identity" section, update the "Bundle Identifier" to match the app's unique identifier.

4. Connect your iOS device or select a simulator in Xcode to run the app.

5. In Xcode, select "Product" > "Archive" to create an archive of the app.

6. In the Organizer window, select the archive and then select "Distribute App" to submit the app to the App Store.

Android:

1. Build the Android app using the react-native run-android command in the project directory.

2. Create a keystore file to sign the app.

3. Build a release APK by running cd android && ./gradlew assembleRelease.

4. Sign the APK using the keystore file created in step 2

5. Submit the app to the Google Play Store using the Google Play Console.

**Preparing for deployment:**

• App Name: Short, catchy, unique, and relevant.

• App Description: Describe what your app is about while trying to populate the description with relevant keywords.

• App Screenshots: Use each screenshot to promote a key feature of your app, with the most important or valuable one first.

• App Version

• Bundle ID: The standard format is com.CompanyName.AppName (e.g. com.spotify.music)

• Developer/Publisher Profile

**App Store Prerequisites**

Some items that you have to prepare specifically for the App Store include:

• App Icon: A 1024x1024 px icon with no transparency.

• Keywords: You have 100 characters to choose as many relevant keywords as you can, separated by commas.

• Countries: Choose whether your app will be available worldwide or in selected countries.

• App Category: Pick an App Store category and subcategory that best suit your app.

• Copyright: YYYY Company Name.

• Test Account: A demo account created to be used by Apple during their review of your app. This account can be removed once your app has been approved.

**Google Play**

• App Icon: A 512x512 px icon. Transparency is allowed.

• App Category: Select one of the Google Play categories.

To publish your Expo app, set the app.json configurations.

{

"expo": {

"name": "App Name",

"icon": "./path/to/your/app-icon.png",

"version": "1.0.0",

"slug": "app-slug",

"sdkVersion": "XX.0.0",

"ios": {

"bundleIdentifier": "com.companyname.appname"

},

"android": {

"package": "com.companyname.appname"

}

}

}

• name: The name of the app as it appears both within Expo and on home screen as a standalone app.

• slug: The URL slug for publishing. For example, "app-name" refers to the expo.io/@username/app-name project.

• sdkVersion: The Expo sdkVersion to run the project. This should line up with the version specified in package.json.

**Step 1: Create an account**

Create the Expo account to publish our application.

https://expo.dev/signup

Once we create an account, we must log in. After we log in, this is what our dashboard should look like:

**Step 2: Deploy application**

Create React Native application and open it in Visual Studio (VS) code. After that, we open the terminal by pressing the Ctrl+` keyboard shortcut.

After this, we log in to the Expo command line interface (CLI) using the terminal. We enter the following command to install Expo CLI:

$ npm install -g expo-cli

If already installed, then ignore.

**Step 3: Logging in**

Log in to the Expo App store through the terminal using the following command:

$ expo login

Enter username and password.

Now , publish React Native application by running the following command:

$ **expo publish**

OR

Building an Android App

1. Run "expo build:android".

[exp] No currently active or previous builds for this project.

? How would you like to upload your credentials?

(Use arrow keys)

❯ Expo handles all credentials, you can still provide overrides

I will provide all the credentials and files needed, Expo does no validation

2. we can either upload own keystore or have Expo generate one for us. If you choose to have Expo generate one , make sure to back it up as we will need it to submit any future updates of the app, and without it we won't be able to update the app anymore.

3. To back up the keystore, run "expo fetch:android:keystore".

4. Android app will now start building. We can check the status of the build by running "expo build:status".

5. Once app has been built, we will be given a .apk URL of Android app that we can download.

Building an iOS App

1. Run "expo build:ios"

2. The same way as Android's keystore, we can either upload own distribution certificate or let Expo handle it.

3. iOS app will now start building. We can check the status of the build by running "expo build:status".

4. Once app has been built, will be given a .ipa url of iOS app that we can download.

Now that you have your .apk and .ipa files ready, we can go ahead and submit them to the Apple App Store and Google Play Store would with any native app.