Deploy React Native Applications Using Fastlane

Deploy React Native Applications Using Fastlane

Prerequisites

Install Fastlane – Again, Fastlane is a lightweight CLI tool for automating the deployment of your mobile application to both Android and iOS platforms.

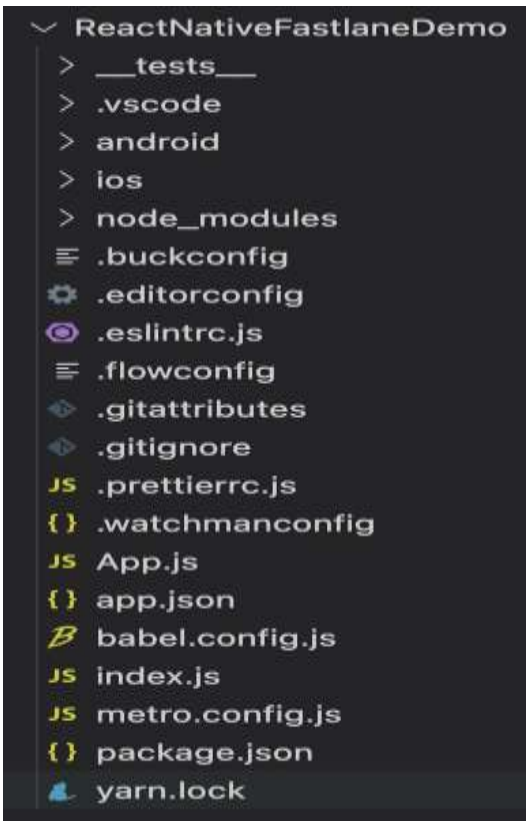
Create a Firebase account – Firebase is a platform, developed by Google, for creating mobile and web applications.

Google Play Developer account – This enables you to publish and manage your apps on the Google Play Store.

Apple Developer account – This enables you to publish and manage your apps on the Apple Store

Directory structure

Your React Native project structure should look like this:



There are two folders you need to take note of: iOS and Android. This is where you need to set up the Fastlane configurations.

Prepare your Android project

Add the following code in the build.gradle in android/app/ folder in the android block.

```
productFlavors {
   internal {}
   production {}
}
```

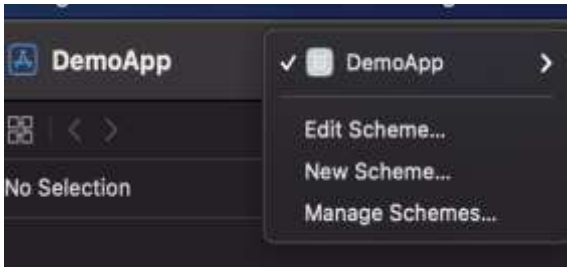
{%callout%} You can refer to this guide on build variants for further configuration options. {%/callout%}

Create a keystore file. If you do not have one, refer here for more information.

Create a google-service-account.json file inside the folder; instructions can be found here.

Prepare your iOS project

1. Execute the workspace project in the iOS folder of the project, this will open up the Xcode.
2. Xcode may ask you to install additional components, select yes to proceed.
3. Once the Xcode is opened, create a new scheme. This is required for deploying a build for different environments.
    a. Click on the project name at the top, and select “New Scheme”.
    b. Input the name of the scheme.



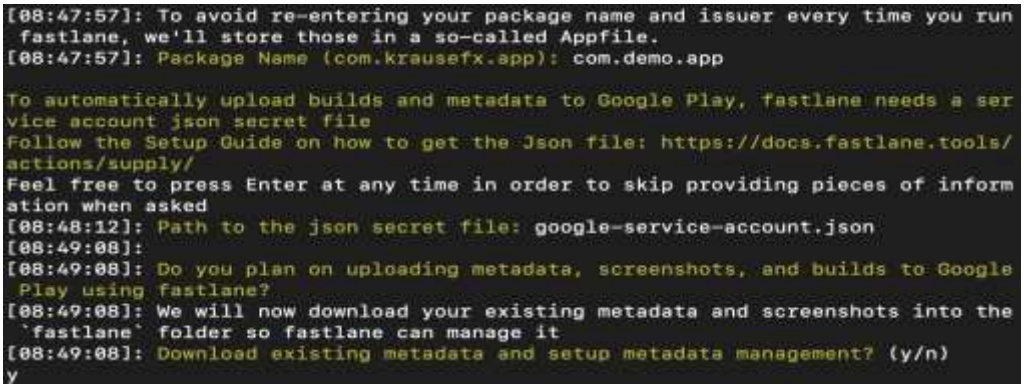
Set up Fastlane

Now you need to set up the automation Fastlane will use to deploy your applications.

1. Change directories to the root of your Android project.
2. Initialize the project with Fastlane.
3. fastlane init

From here, Fastlane will automatically detect your project, and ask for any missing information.

Sample Fastlane output



Note: **Repeat the steps in this section for your iOS project.**

Once the setup has finished you can see a new folder inside both platform root directory.

- fastlane/

- Appfile

- Fastfile

For information:

Appfile contains identifiers used to connect to the Developer Portal and App Store Connect. You can read more about this file here.

Fastfile contains all actions you can launch. You can read more about this file here.

The Fastlane scripts will be placed in the root of each of your project directories. Below is our some snippets.

Android Fastfile

```
default_platform(:android)

platform :android do
  desc "Deploy app for internal sharing"
  lane :deploy_internal do
    ENV["ENVFILE"] = "../env/.env"

    gradle(
      task: "clean"
    )

    android_set_version_code

    gradle(
      task: "bundle",
      flavor: "internal",
```

```
      build_type: "Release"
    )

    upload_to_play_store(
      track: 'internal',
      skip_upload_images: true,
      skip_upload_screenshots: true,
      version_code: android_get_version_code,
      version_name: android_get_version_name
    )
  end

  desc "Deploy app to Production"
  lane :deploy_production do
    ENV["ENVFILE"] = "../env/.env.prod"

    gradle(
      task: "clean"
    )

    android_set_version_code
    android_set_version_name(bump_type: "patch")

    gradle(
      task: "bundle",
      flavor: "production",
      build_type: "Release"
    )

    upload_to_play_store(
      release_status: "draft",
      skip_upload_images: true,
      skip_upload_screenshots: true,
      version_code: android_get_version_code,
      version_name: android_get_version_name
    )
 end
end
```

iOS Fastfile

```
efault_platform(:ios)

platform :ios do
  desc "Deploy app for internal sharing"
  lane :deploy_internal do
    ENV["ENVFILE"] = "DEV"
    cocoapods(
      clean_install: true
    )
    increment_build_number
    cert
```

```
    sigh
    gym(scheme: "DemoApp")
    upload_to_testflight
  end

  desc "Deploy app to Production"
  lane :deploy_production do
    get_push_certificate
    ENV["ENVFILE"] = "PROD"
    cocoapods(
      clean_install: true
    )
    increment_version_number(bump_type: "patch")
    increment_build_number(build_number: "0")
    cert
    sigh
    gym(scheme: "DemoApp (Prod)")
    upload_to_app_store(ignore_language_directory_validation: true, force: true)
  end
end
```

Deploy your applications

The deployment pipelines for your Android and iOS applications have now been configured.

Deploy your Android build

Deploy your Android application to your “Dev” target. This way, you can validate functionality prior to pushing to production.

Fastlane <Lane_Name>

fastlane deploy_internal

Once you are satisfied with the results in “Dev”, go ahead and release “Prod”.

fastlane deploy_production

Deploy your iOS build

In a very similar way as you tested your Android project, deploy your iOS project first to your “Dev” environment.

Fastlane <Lane_Name>

fastlane deploy_internal

And finally, once you are satisfied, deploy to “Prod”.

fastlane deploy_production

Now we have a fully automated deployment for your Android and iOS projects.