**Push Notification in IOS**

1. Install package namely:-

@react-native-community/push-notification-ios

@react-native-firebase/app

@react-native-firebase/messaging

react-native-push-notification

**Note:** Check the documentation of @react-native-community/push-notification-ios npm and follow the instruction strictly in the git repository to avoid errors.

1. Add line in PodFile:-

Add line pod 'FirebaseCore', :modular\_headers => true (Right after **target <ProjectName> do**)

Ie,

target **<ProjectName>** do

pod 'FirebaseCore', :modular\_headers => true

….

Also add lines:-

….

pod 'BVLinearGradient', :path => '../node\_modules/react-native-linear-gradient'

permissions\_path = '../node\_modules/react-native-permissions/ios'

pod 'Permission-Camera', :path => "#{permissions\_path}/Camera"

pod 'Permission-PhotoLibrary', :path => "#{permissions\_path}/PhotoLibrary"

pod 'RNVectorIcons', :path => '../node\_modules/react-native-vector-icons'

pod 'Permission-LocationAlways', :path => "#{permissions\_path}/LocationAlways"

pod 'Permission-LocationWhenInUse', :path => "#{permissions\_path}/LocationWhenInUse"

pod 'Firebase', :modular\_headers => true

pod 'FirebaseCoreInternal', :modular\_headers => true

pod 'GoogleUtilities', :modular\_headers => true

pod "Firebase/Messaging"

pod 'RNFBApp', :path => '../node\_modules/@react-native-firebase/app'

pod 'RNFBMessaging', :path => '../node\_modules/@react-native-firebase/messaging'

pod 'RNCPushNotificationIOS', :path => '../node\_modules/@react-native-community/push-notification-ios'

target 'FitCheckTests' do

….

### After the changes in Podfile ,AppDelegate.h and AppDelegate.m run the command in project folder cd ios && pod install

3.Add line In AddDelegate.h

#import <React/RCTBridgeDelegate.h>

#import <UIKit/UIKit.h>

#import <UserNotifications/UNUserNotificationCenter.h>

@interface AppDelegate : UIResponder <UIApplicationDelegate, RCTBridgeDelegate,UNUserNotificationCenterDelegate>

@property (nonatomic, strong) UIWindow \*window;

@end

4.Add line In AddDelegate.m

#import "AppDelegate.h"

#import <React/RCTBridge.h>

#import <React/RCTBundleURLProvider.h>

#import <React/RCTRootView.h>

#import <Firebase.h>

#import <UserNotifications/UserNotifications.h>

#import <RNCPushNotificationIOS.h>

#import <React/RCTLinkingManager.h>

#ifdef FB\_SONARKIT\_ENABLED

#import <FlipperKit/FlipperClient.h>

#import <FlipperKit/FlipperClient.h>

#import <FlipperKitLayoutPlugin/FlipperKitLayoutPlugin.h>

#import <FlipperKitUserDefaultsPlugin/FKUserDefaultsPlugin.h>

#import <FlipperKitNetworkPlugin/FlipperKitNetworkPlugin.h>

#import <SKIOSNetworkPlugin/SKIOSNetworkAdapter.h>

#import <FlipperKitReactPlugin/FlipperKitReactPlugin.h>

static void InitializeFlipper(UIApplication \*application) {

FlipperClient \*client = [FlipperClient sharedClient];

SKDescriptorMapper \*layoutDescriptorMapper = [[SKDescriptorMapper alloc] initWithDefaults];

[client addPlugin:[[FlipperKitLayoutPlugin alloc] initWithRootNode:application withDescriptorMapper:layoutDescriptorMapper]];

[client addPlugin:[[FKUserDefaultsPlugin alloc] initWithSuiteName:nil]];

[client addPlugin:[FlipperKitReactPlugin new]];

[client addPlugin:[[FlipperKitNetworkPlugin alloc] initWithNetworkAdapter:[SKIOSNetworkAdapter new]]];

[client start];

}

#endif

@implementation AppDelegate

- (BOOL)application:(UIApplication \*)application didFinishLaunchingWithOptions:(NSDictionary \*)launchOptions

{

[GMSServices provideAPIKey:@"AIzaSyCZ9kuVUyhZxeFR3cPnebauMlffVOhoM1Y"]; // add this line using the api key obtained from Google Console

if ([FIRApp defaultApp] == nil) {

[FIRApp configure];

}

#ifdef FB\_SONARKIT\_ENABLED

InitializeFlipper(application);

#endif

RCTBridge \*bridge = [[RCTBridge alloc] initWithDelegate:self launchOptions:launchOptions];

RCTRootView \*rootView = [[RCTRootView alloc] initWithBridge:bridge

moduleName:@"Phil"

initialProperties:nil];

rootView.backgroundColor = [[UIColor alloc] initWithRed:1.0f green:1.0f blue:1.0f alpha:1];

self.window = [[UIWindow alloc] initWithFrame:[UIScreen mainScreen].bounds];

UIViewController \*rootViewController = [UIViewController new];

rootViewController.view = rootView;

self.window.rootViewController = rootViewController;

[self.window makeKeyAndVisible];

UNUserNotificationCenter \*center = [UNUserNotificationCenter currentNotificationCenter];

center.delegate = self;

return YES;

}

-(void)userNotificationCenter:(UNUserNotificationCenter \*)center willPresentNotification:(UNNotification \*)notification withCompletionHandler:(void (^)(UNNotificationPresentationOptions options))completionHandler

{

completionHandler(UNNotificationPresentationOptionSound | UNNotificationPresentationOptionAlert | UNNotificationPresentationOptionBadge);

}

- (NSURL \*)sourceURLForBridge:(RCTBridge \*)bridge

{

#if DEBUG

return [[RCTBundleURLProvider sharedSettings] jsBundleURLForBundleRoot:@"index" fallbackResource:nil];

#else

return [[NSBundle mainBundle] URLForResource:@"main" withExtension:@"jsbundle"];

#endif

}

// Required for the register event.

- (void)application:(UIApplication \*)application didRegisterForRemoteNotificationsWithDeviceToken:(NSData \*)deviceToken

{

[RNCPushNotificationIOS didRegisterForRemoteNotificationsWithDeviceToken:deviceToken];

}

// Required for the notification event. You must call the completion handler after handling the remote notification.

- (void)application:(UIApplication \*)application didReceiveRemoteNotification:(NSDictionary \*)userInfo

fetchCompletionHandler:(void (^)(UIBackgroundFetchResult))completionHandler

{

[RNCPushNotificationIOS didReceiveRemoteNotification:userInfo fetchCompletionHandler:completionHandler];

}

// Required for the registrationError event.

- (void)application:(UIApplication \*)application didFailToRegisterForRemoteNotificationsWithError:(NSError \*)error

{

[RNCPushNotificationIOS didFailToRegisterForRemoteNotificationsWithError:error];

}

// Required for localNotification event

- (void)userNotificationCenter:(UNUserNotificationCenter \*)center

didReceiveNotificationResponse:(UNNotificationResponse \*)response

withCompletionHandler:(void (^)(void))completionHandler

{

[RNCPushNotificationIOS didReceiveNotificationResponse:response];

}

- (BOOL)application:(UIApplication \*)application openURL:(NSURL \*)url

sourceApplication:(NSString \*)sourceApplication annotation:(id)annotation

{

return [RCTLinkingManager application:application openURL:url

sourceApplication:sourceApplication annotation:annotation];

}

@end

Note: Before this process pls add ios project in firebase account and add APNs Authentication Key in the ios app. IT is in Project Settings => Cloud Messaging =>Apple app configuration option => generate APNs Authentication Key.

To generate APNs Authentication Key u need .p8 file from Apple Developer Account ,Key Id(mention in the naming of the .p8 file) and Team ID(mention in Membership details in Apple Developer Account )