TÀI LIỆU HƯỚNG DẪN TÍCH HỢP SDK REACT NATIVE

EKYC

I. Tích hợp SDK vào iOS

1. Yêu cầu

- Yêu cầu có sẵn 1 dự án **React Native**
- Mở thư mục ios trong dự án **React Native** và tích hợp SDK vào dự án theo các bước của tài liệu (phần 1 và 2 trong mục **II. Tích hợp thư viện SDK vào dự án**):

  iOS VNPT_eKYC_Tutorial_Integrated v3.2.2 [Tiếng Việt]

2. Thêm cầu nối

- Mở dự án iOS bằng Xcode và thêm tệp header (EkycBridgeModule.h) và implementation (EkycBridgeModule.m)

```
#ifndef EkycBridgeModule_h
    #define EkycBridgeModule_h
    #import <React/RCTBridgeModule.h>
    #import "ICSdkEKYC/ICSdkEKYC.h"
    @interface EkycBridgeModule: NSObject <RCTBridgeModule,
    ICMainNFCReaderDelegate>
    @property(nonatomic, copy) RCTPromiseResolveBlock _resolve;
    @property(nonatomic, copy) RCTPromiseRejectBlock _reject;
    @end
    #endif /* EkycBridgeModule_h */
```

- Ở tệp implementation:

```
#import <Foundation/Foundation.h>
#import "EkycBridgeModule.h"
#import "ICSdkEKYC/ICSdkEKYC.h"
@implementation EkycBridgeModule
// To export a module named RCTCalendarModule
RCT_EXPORT_MODULE(EkycBridge);
RCT_EXPORT_METHOD(startEkycFull:(RCTPromiseResolveBlock)resolve
rejecter:(RCTPromiseRejectBlock)reject) {
  NSLog(@"Hello world");
```

```
  self._resolve = resolve;
  self._reject = reject;

   /**** cài đặt ICEkycCameraViewController như tích hợp native ***/

  dispatch_async(dispatch_get_main_queue(), ^{
    UIViewController *root = [[[UIApplication sharedApplication] delegate]
window].rootViewController;
    BOOL modalPresent = (BOOL) (root.presentedViewController);

    if (modalPresent) {
      UIViewController *parent = root.presentedViewController;
      [parent setModalPresentationStyle:UIModalPresentationFullScreen];
      [parent showViewController:camera sender:parent];

    } else {
      [camera setModalPresentationStyle:UIModalPresentationFullScreen];
      [root showDetailViewController:camera sender:root];
    }

  });
};
RCT_EXPORT_METHOD(startEkycOcr:(RCTPromiseResolveBlock)resolve
rejecter:(RCTPromiseRejectBlock)reject) {
  NSLog(@"Hello world");

  self._resolve = resolve;
  self._reject = reject;

 /**** cài đặt ICEkycCameraViewController như tích hợp native ***/

  dispatch_async(dispatch_get_main_queue(), ^{
    UIViewController *root = [[[UIApplication sharedApplication] delegate]
window].rootViewController;
    BOOL modalPresent = (BOOL) (root.presentedViewController);

    if (modalPresent) {
      UIViewController *parent = root.presentedViewController;
      [parent setModalPresentationStyle:UIModalPresentationFullScreen];
      [parent showViewController:camera sender:parent];

    } else {
      [camera setModalPresentationStyle:UIModalPresentationFullScreen];
      [root showDetailViewController:camera sender:root];
    }

  });
};
RCT_EXPORT_METHOD(startEkycFace:(RCTPromiseResolveBlock)resolve
rejecter:(RCTPromiseRejectBlock)reject) {
  NSLog(@"Hello world");
```

```
  self._resolve = resolve;
  self._reject = reject;

 /**** cài đặt ICEkycCameraViewController như tích hợp native ***/


  dispatch_async(dispatch_get_main_queue(), ^{
    UIViewController *root = [[[UIApplication sharedApplication] delegate]
window].rootViewController;
    BOOL modalPresent = (BOOL) (root.presentedViewController);

    if (modalPresent) {
      UIViewController *parent = root.presentedViewController;
      [parent setModalPresentationStyle:UIModalPresentationFullScreen];
      [parent showViewController:camera sender:parent];

    } else {
      [camera setModalPresentationStyle:UIModalPresentationFullScreen];
      [root showDetailViewController:camera sender:root];
    }

  });
};
```

- Trong tệp mẫu cần chú ý đến các hàm **RCT_EXPORT_METHOD**, đây là các method định nghĩa tương ứng với hàm bên code react (**startEkycFull**, **startEkycOcr**, **startEkycFace**) có thể tạo nhiều hơn
    - **startEkycFull**: mở SDK thực hiện luồng đầy đủ (thẻ và khuôn mặt)
    - **startEkycOcr**: mở SDK thực hiên luồng quét thẻ
    - **startEkycFace**: mở SDK thực hiện luồng xác thực khuôn mặt
- Các phương thức con lại (nhận kết quả qua **Delegate**) tương tự tích hợp với dự án Native iOS (đề cập ở phần 1)

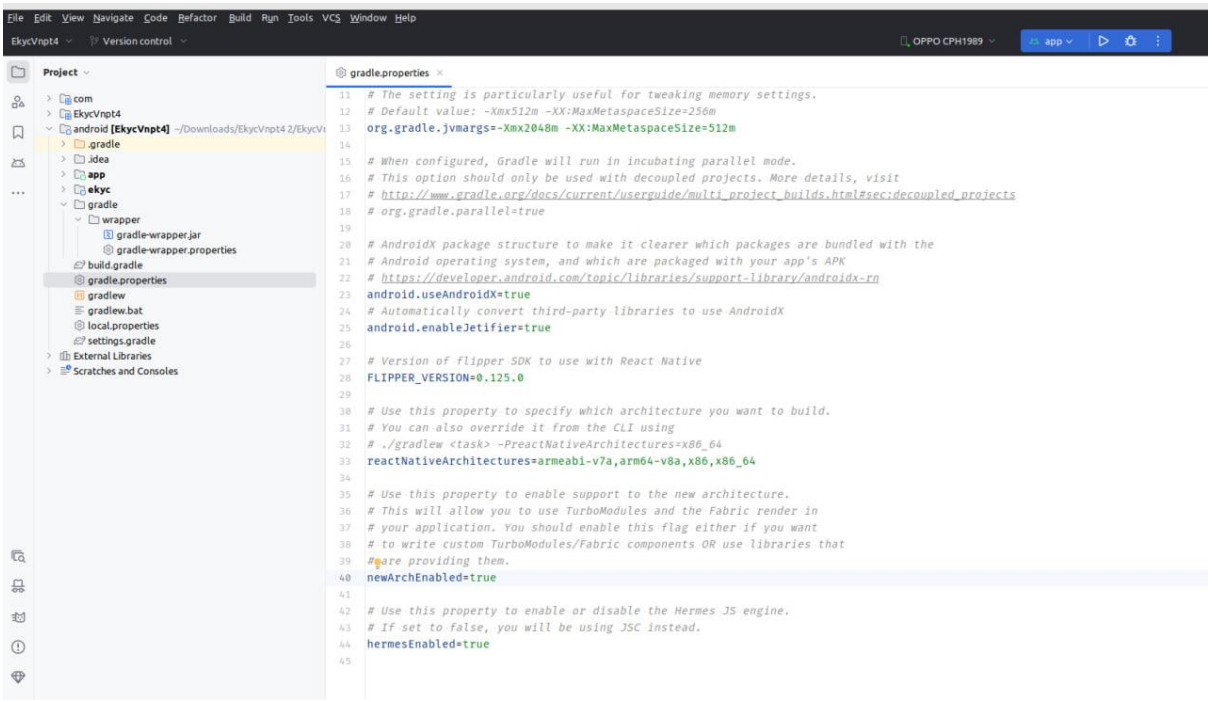
II. Tích hợp SDK vào Android

1. Yêu cầu

- Yêu cầu đã có sẵn một dự án **React Native**
- Mở thư mục android trong dự án **React Native** và tích hợp theo các bước theo tài liệu sau: Android VNPT_eKYC_Tutorial_Integrated v3.3.3
- Nếu sử dụng **React Native > v0.70** thì sẽ config như sau:

```
android {
    ndkVersion "23.1.7779620"
```

- Sử dụng ndk version 23.1.7779620 trong build.gradle (app)



- Enable newArchEnabled=true trong gradle.properties

2. Thêm cầu nối

- Tạo file cầu nối **React Native (EkycBridgeModule.java)** kế thừa từ **ReactContextBaseJavaModule** như sau:

```
import android.app.Activity;
import android.app.Activity;
import android.content.Intent;
```

```
import androidx.annotation.NonNull;

import com.facebook.react.bridge.ActivityEventListener;
import com.facebook.react.bridge.BaseActivityEventListener;
import com.facebook.react.bridge.Promise;
import com.facebook.react.bridge.ReactApplicationContext;
import com.facebook.react.bridge.ReactContextBaseJavaModule;
import com.facebook.react.bridge.ReactMethod;
import com.google.gson.JsonObject;
import com.google.gson.JsonParser;
import com.vnptit.idg.sdk.activity.VnptIdentityActivity;
import com.vnptit.idg.sdk.activity.VnptOcrActivity;
import com.vnptit.idg.sdk.activity.VnptPortraitActivity;
import com.vnptit.idg.sdk.utils.KeyIntentConstants;
import com.vnptit.idg.sdk.utils.KeyResultConstants;
import com.vnptit.idg.sdk.utils.SDKEnum;

public class EkycBridgeModule extends ReactContextBaseJavaModule {
  private static final int EKYC_REQUEST_CODE = 100;
  private static final String LOG_OCR = "LOG_OCR";
  private static final String LOG_LIVENESS_CARD_FRONT =
"LOG_LIVENESS_CARD_RESULT";
  private static final String LOG_LIVENESS_CARD_REAR =
"LOG_LIVENESS_CARD_REAR";
  private static final String LOG_COMPARE = "LOG_COMPARE";
  private static final String LOG_LIVENESS_FACE = "LOG_LIVENESS_FACE";
  private static final String LOG_MASK_FACE = "LOG_MASK_FACE";

  private Promise mEkycPromise;

  public EkycBridgeModule(ReactApplicationContext reactContext) {
     super(reactContext);

     final ActivityEventListener activityEventListener = new
BaseActivityEventListener() {
        @Override
        public void onActivityResult(Activity activity, int requestCode, int
resultCode, Intent data) {
           if (requestCode == EKYC_REQUEST_CODE && resultCode ==
Activity.RESULT_OK) {
              if (data != null &&  mEkycPromise != null) {
                 final String dataInfoResult =
data.getStringExtra(KeyResultConstants.INFO_RESULT);
                 final String dataLivenessCardFrontResult =
data.getStringExtra(KeyResultConstants.LIVENESS_CARD_FRONT_RESULT);
                 final String dataLivenessCardRearResult =
data.getStringExtra(KeyResultConstants.LIVENESS_CARD_REAR_RESULT);
                 final String dataCompareResult =
```

```
data.getStringExtra(KeyResultConstants.COMPARE_RESULT);
                final String dataLivenessFaceResult =
data.getStringExtra(KeyResultConstants.LIVENESS_FACE_RESULT);
                final String dataMaskedFaceResult =
data.getStringExtra(KeyResultConstants.MASKED_FACE_RESULT);

                final JsonObject json = new JsonObject();
                json.addProperty(LOG_OCR, dataInfoResult);
                json.addProperty(LOG_LIVENESS_CARD_FRONT,
dataLivenessCardFrontResult);
                json.addProperty(LOG_LIVENESS_CARD_REAR,
dataLivenessCardRearResult);
                json.addProperty(LOG_COMPARE, dataCompareResult);
                json.addProperty(LOG_LIVENESS_FACE,
dataLivenessFaceResult);
                json.addProperty(LOG_MASK_FACE, dataMaskedFaceResult);
                mEkycPromise.resolve(json.toString());
            }

            mEkycPromise = null;
          }
        }
    };
    reactContext.addActivityEventListener(activityEventListener);
  }

  @NonNull
  @Override
  public String getName() {
    return "EkycBridge";
  }

  // Phương thức thực hiện eKYC luồng đầy đủ bao gồm: Chụp ảnh giấy tờ và
chụp ảnh chân dung
  // Bước 1 - chụp ảnh chân dung xa gần
  // Bước 2 - hiển thị kết quả
  @ReactMethod
  private void startEkycFace(final String args, final Promise promise) {
    final Activity currentActivity = getCurrentActivity();
    if (currentActivity == null) {
      return;
    }

    mEkycPromise = promise;

    final Intent intent = getBaseIntent(args, currentActivity,
VnptPortraitActivity.class);
```

```
    // Giá trị này xác định phiên bản khi sử dụng Máy ảnh tại bước chụp ảnh
chân dung luồng full. Mặc định là Normal ✓
    // - Normal: chụp ảnh chân dung 1 hướng
    // - ADVANCED: chụp ảnh chân dung xa gần
    intent.putExtra(KeyIntentConstants.VERSION_SDK,
SDKEnum.VersionSDKEnum.ADVANCED.getValue());

    // Bật/[Tắt] chức năng So sánh ảnh trong thẻ và ảnh chân dung
    intent.putExtra(KeyIntentConstants.IS_COMPARE_FLOW, false);

    // Bật/Tắt chức năng kiểm tra che mặt
    intent.putExtra(KeyIntentConstants.IS_CHECK_MASKED_FACE, true);

    // Lựa chọn chức năng kiểm tra ảnh chân dung chụp trực tiếp (liveness
face)
    // - NoneCheckFace: Không thực hiện kiểm tra ảnh chân dung chụp trực
tiếp hay không
    // - IBeta: Kiểm tra ảnh chân dung chụp trực tiếp hay không iBeta
(phiên bản hiện tại)
    // - Standard: Kiểm tra ảnh chân dung chụp trực tiếp hay không Standard
(phiên bản mới)
    intent.putExtra(KeyIntentConstants.CHECK_LIVENESS_FACE,
SDKEnum.ModeCheckLiveNessFace.iBETA.getValue());

    currentActivity.startActivityForResult(intent, EKYC_REQUEST_CODE);
  }


  // Phương thức thực hiện eKYC luồng "Chụp ảnh giấy tờ"
  // Bước 1 - chụp ảnh giấy tờ
  // Bước 2 - hiển thị kết quả
  @ReactMethod
  private void startEkycOcr(final String args, final Promise promise) {
    final Activity currentActivity = getCurrentActivity();
    if (currentActivity == null) {
      return;
    }

    mEkycPromise = promise;

    final Intent intent = getBaseIntent(args, currentActivity,
VnptOcrActivity.class);

    // Giá trị này xác định kiểu giấy tờ để sử dụng:
    // - IdentityCard: Chứng minh thư nhân dân, Căn cước công dân
    // - IDCardChipBased: Căn cước công dân gắn Chip
    // - Passport: Hộ chiếu
    // - DriverLicense: Bằng lái xe
```

```
    // - MilitaryIdCard: Chứng minh thư quân đội
    intent.putExtra(KeyIntentConstants.DOCUMENT_TYPE,
SDKEnum.DocumentTypeEnum.IDENTITY_CARD.getValue());

    // Bật/Tắt chức năng kiểm tra ảnh giấy tờ chụp trực tiếp (liveness
card)
    intent.putExtra(KeyIntentConstants.IS_CHECK_LIVENESS_CARD, true);

    // Lựa chọn chế độ kiểm tra ảnh giấy tờ ngay từ SDK
    // - None: Không thực hiện kiểm tra ảnh khi chụp ảnh giấy tờ
    // - Basic: Kiểm tra sau khi chụp ảnh
    // - MediumFlip: Kiểm tra ảnh hợp lệ trước khi chụp (lật giấy tờ thành
công → hiển thị nút chụp)
    // - Advance: Kiểm tra ảnh hợp lệ trước khi chụp (hiển thị nút chụp)
    intent.putExtra(KeyIntentConstants.TYPE_VALIDATE_DOCUMENT,
SDKEnum.TypeValidateDocument.Basic.getValue());

    currentActivity.startActivityForResult(intent, EKYC_REQUEST_CODE);
  }

  // Phương thức thực hiện eKYC luồng đầy đủ bao gồm: Chụp ảnh giấy tờ và
chụp ảnh chân dung
  // Bước 1 - chụp ảnh giấy tờ
  // Bước 2 - chụp ảnh chân dung xa gần
  // Bước 3 - hiển thị kết quả
  @ReactMethod
  private void startEkycFull(final String args, final Promise promise) {
     final Activity currentActivity = getCurrentActivity();
     if (currentActivity == null) {
        return;
     }

     mEkycPromise = promise;

     final Intent intent = getBaseIntent(args, currentActivity,
VnptIdentityActivity.class);

     // Giá trị này xác định kiểu giấy tờ để sử dụng:
     // - IDENTITY_CARD: Chứng minh thư nhân dân, Căn cước công dân
     // - IDCardChipBased: Căn cước công dân gắn Chip
     // - Passport: Hộ chiếu
     // - DriverLicense: Bằng lái xe
     // - MilitaryIdCard: Chứng minh thư quân đội
     intent.putExtra(KeyIntentConstants.DOCUMENT_TYPE,
SDKEnum.DocumentTypeEnum.IDENTITY_CARD.getValue());

     // Bật/Tắt chức năng So sánh ảnh trong thẻ và ảnh chân dung
     intent.putExtra(KeyIntentConstants.IS_COMPARE_FLOW, true);
```

```
    // Bật/Tắt chức năng kiểm tra ảnh giấy tờ chụp trực tiếp (liveness
card)
    intent.putExtra(KeyIntentConstants.IS_CHECK_LIVENESS_CARD, true);

    // Lựa chọn chức năng kiểm tra ảnh chân dung chụp trực tiếp (liveness
face)
    // - NoneCheckFace: Không thực hiện kiểm tra ảnh chân dung chụp trực
tiếp hay không
    // - iBETA: Kiểm tra ảnh chân dung chụp trực tiếp hay không iBeta
(phiên bản hiện tại)
    // - Standard: Kiểm tra ảnh chân dung chụp trực tiếp hay không Standard
(phiên bản mới)
    intent.putExtra(KeyIntentConstants.CHECK_LIVENESS_FACE,
SDKEnum.ModeCheckLiveNessFace.iBETA.getValue());

    // Bật/Tắt chức năng kiểm tra che mặt
    intent.putExtra(KeyIntentConstants.IS_CHECK_MASKED_FACE, true);

    // Lựa chọn chế độ kiểm tra ảnh giấy tờ ngay từ SDK
    // - None: Không thực hiện kiểm tra ảnh khi chụp ảnh giấy tờ
    // - Basic: Kiểm tra sau khi chụp ảnh
    // - MediumFlip: Kiểm tra ảnh hợp lệ trước khi chụp (lật giấy tờ thành
công → hiển thị nút chụp)
    // - Advance: Kiểm tra ảnh hợp lệ trước khi chụp (hiển thị nút chụp)
    intent.putExtra(KeyIntentConstants.TYPE_VALIDATE_DOCUMENT,
SDKEnum.TypeValidateDocument.Basic.getValue());

    // Giá trị này xác định việc có xác thực số ID với mã tỉnh thành, quận
huyện, xã phường tương ứng hay không.
    intent.putExtra(KeyIntentConstants.IS_VALIDATE_POSTCODE, true);

    // Giá trị này xác định phiên bản khi sử dụng Máy ảnh tại bước chụp ảnh
chân dung luồng full. Mặc định là Normal ✓
    // - Normal: chụp ảnh chân dung 1 hướng
    // - ProOval: chụp ảnh chân dung xa gần
    intent.putExtra(KeyIntentConstants.VERSION_SDK,
SDKEnum.VersionSDKEnum.ADVANCED.getValue());

    currentActivity.startActivityForResult(intent, EKYC_REQUEST_CODE);
  }

  private Intent getBaseIntent(final String args, final Activity activity,
final Class<?> clazz) {
    final Intent intent = new Intent(activity, clazz);

    final JsonObject json = JsonParser.parseString(args).getAsJsonObject();
```

```
        // Nhập thông tin bộ mã truy cập. Lấy tại mục Quản lý Token
https://ekyc.vnpt.vn/admin-dashboard/console/project-manager
        intent.putExtra(KeyIntentConstants.ACCESS_TOKEN,
json.get("access_token").getAsString());
        intent.putExtra(KeyIntentConstants.TOKEN_ID,
json.get("token_id").getAsString());
        intent.putExtra(KeyIntentConstants.TOKEN_KEY,
json.get("token_key").getAsString());

        // Giá trị này dùng để đảm bảo mỗi yêu cầu (request) từ phía khách hàng
sẽ không bị thay đổi.
        intent.putExtra(KeyIntentConstants.CHALLENGE_CODE, "INNOVATIONCENTER");

        // Ngôn ngữ sử dụng trong SDK
        // - VIETNAMESE: Tiếng Việt
        // - ENGLISH: Tiếng Anh
        intent.putExtra(KeyIntentConstants.LANGUAGE_SDK,
SDKEnum.LanguageEnum.VIETNAMESE.getValue());

        // Bật/Tắt Hiển thị màn hình hướng dẫn
        intent.putExtra(KeyIntentConstants.IS_SHOW_TUTORIAL, true);

        // Bật chức năng hiển thị nút bấm "Bỏ qua hướng dẫn" tại các màn hình
hướng dẫn bằng video
        intent.putExtra(KeyIntentConstants.IS_ENABLE_GOT_IT, true);

        // Sử dụng máy ảnh mặt trước
        // - FRONT: Camera trước
        // - BACK: Camera trước
        intent.putExtra(KeyIntentConstants.CAMERA_POSITION_FOR_PORTRAIT,
SDKEnum.CameraTypeEnum.FRONT.getValue());

        return intent;
    }
}
```

- Tạo một file kế thừa từ **ReactPackage (EkycBridgeReactPackage.java)** khai báo cầu nối **React Native (EkycBridgeModule.java)** vừa khởi tạo trong native module (hàm **createNativeModules**) như sau:

```
import androidx.annotation.NonNull;

import com.facebook.react.ReactPackage;
import com.facebook.react.bridge.NativeModule;
import com.facebook.react.bridge.ReactApplicationContext;
import com.facebook.react.uimanager.ViewManager;
```

```
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

public class EkycBridgeReactPackage implements ReactPackage {
  @NonNull
  @Override
  public List<NativeModule> createNativeModules(@NonNull
ReactApplicationContext reactApplicationContext) {
     List<NativeModule> modules = new ArrayList<>();
     modules.add(new EkycBridgeModule(reactApplicationContext));
     return modules;
  }

  @NonNull
  @Override
  public List<ViewManager> createViewManagers(@NonNull
ReactApplicationContext reactApplicationContext) {
     return Collections.emptyList();
  }
}
```

- Thêm package **EkycBridgeReactPackage** vừa tạo vào list packages của **ReactNativeHost** trong file **MainApplication.java** như sau:

```
import android.app.Application;
import com.facebook.react.PackageList;
import com.facebook.react.ReactApplication;
import com.facebook.react.ReactNativeHost;
import com.facebook.react.ReactPackage;
import com.facebook.react.defaults.DefaultNewArchitectureEntryPoint;
import com.facebook.react.defaults.DefaultReactNativeHost;
import com.facebook.soloader.SoLoader;

import java.util.List;

public class MainApplication extends Application implements ReactApplication
{

 private final ReactNativeHost mReactNativeHost =
     new DefaultReactNativeHost(this) {
       @Override
       public boolean getUseDeveloperSupport() {
         return BuildConfig.DEBUG;
       }
```

```
      @Override
      protected List<ReactPackage> getPackages() {
        @SuppressWarnings("UnnecessaryLocalVariable")
        List<ReactPackage> packages = new PackageList(this).getPackages();
        // Packages that cannot be autolinked yet can be added manually
here, for example:
         packages.add(new EkycBridgeReactPackage());
        return packages;
      }

      @Override
      protected String getJSMainModuleName() {
        return "index";
      }

      @Override
      protected boolean isNewArchEnabled() {
        return BuildConfig.IS_NEW_ARCHITECTURE_ENABLED;
      }

      @Override
      protected Boolean isHermesEnabled() {
        return BuildConfig.IS_HERMES_ENABLED;
      }
    };

 @Override
 public ReactNativeHost getReactNativeHost() {
   return mReactNativeHost;
 }

 @Override
 public void onCreate() {
   super.onCreate();
   SoLoader.init(this, /* native exopackage */ false);
   if (BuildConfig.IS_NEW_ARCHITECTURE_ENABLED) {
     // If you opted-in for the New Architecture, we load the native entry
point for this app.
     DefaultNewArchitectureEntryPoint.load();
   }
   ReactNativeFlipper.initializeFlipper(this,
getReactNativeHost().getReactInstanceManager());
 }
}
```

- Khai báo **MainApplication** trong file **AndroidManifest.xml** nếu chưa khai báo như sau:

AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android">

    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:name=".MainApplication"
        android:label="SmartUXIntegrateRN"
        android:icon="@mipmap/ic_launcher"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:allowBackup="false"
        android:theme="@style/AppTheme">
        <activity
            android:name=".MainActivity"
            android:label="SmartUXIntegrateRN"
            android:configChanges="keyboard|keyboardHidden|orientation|screenLayout|screenSize|smallestScreenSize|uiMode"
            android:launchMode="singleTask"
            android:windowSoftInputMode="adjustResize"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

III. Xây dựng cầu nối

1. Xây dựng các file cầu nối

- Tạo file **SDKEkyc.js** để làm cầu nối tương tác giữa **React Native** và **iOS**, **Android**

```
import {NativeModules} from "react-native";

const {EkycBridge} = NativeModules;
const SDKEkyc = {};

SDKEkyc.startEkycFull = async function () {
try {
const json = JSON.stringify({
access_token: '<ACCESS_TOKEN> (including bearer)',
token_id: '<TOKEN_ID>',
token_key: '<TOKEN_KEY>',
});
return await EkycBridge.startEkycFull(json)
} catch (e) {
return {
'error': e.message
}
}
};

SDKEkyc.startEkycOcr = async function () {
try {
const json = JSON.stringify({
access_token: '<ACCESS_TOKEN> (including bearer)',
token_id: '<TOKEN_ID>',
token_key: '<TOKEN_KEY>',
});
return await EkycBridge.startEkycOcr(json)
} catch (e) {
return {
'error': e.message
}
}
};

SDKEkyc.startEkycFace = async function () {
try {
const json = JSON.stringify({
access_token: '<ACCESS_TOKEN> (including bearer)',
token_id: '<TOKEN_ID>',
```

```
token_key: '<TOKEN_KEY>',
});
return await EkycBridge.startEkycFace(json)
} catch (e) {
return {
'error': e.message
}
}
};

export default SDKEkyc;
```

2. Ví dụ

```
// sử dụng luồng full
const response = await SDKEkyc.startEkycFull();
if (response.error) {
  //  xử lý trường hợp lỗi
} else {
 // xử lý trường hợp thành công với dữ liệu `response`
}
```