Easebuzz React Native Integration

Download React Native SDK from github using the link below, extract it and put it in the relevant directory.

https://github.com/easebuzz/paywitheasebuzz-react-native-lib

Installing SDK:

1. Run the below commands to install sdk into your project.

1.1 npm install $(npm pack <Path of React Native SDK>/react-native-easebuzz-kit | tail -1)

Example : npm install $(npm pack HomeDirectory/SDKfolder/react-native-easebuzz-kit | tail -1)

For React Native 0.59 and lower

1.2. react-native link react-native-easebuzz-kit

Android Setup:

1. Copy **peb-lib.aar** file into android/app/libs/ folder of your react native application(If libs folder is not there, Please create it manually).

1.1. **Build.gradle(app) modifications.**

Add this line to build.gradle (app)

```
defaultConfig {
        multiDexEnabled true
    }
```

Add the following lines to packagingOptions,

```
exclude 'META-INF/DEPENDENCIES'
exclude 'META-INF/NOTICE'
exclude 'META-INF/LICENSE'
exclude 'META-INF/LICENSE.txt'
exclude 'META-INF/NOTICE.txt'
```

Add the following line to dexOptions.

```
javaMaxHeapSize "4g"
```

Add repositories section as follows.

```
repositories {
    flatDir {
        dirs 'libs'
        }
    }
```

Add the following dependencies

```
implementation (name: 'peb-lib', ext: 'aar')
implementation 'com.android.support:appcompat-v7:28.0.0'
implementation 'com.android.support:design:28.0.0'
```

```
implementation 'com.android.support:multidex:1.0.1'
implementation 'com.squareup.okhttp:okhttp:2.4.0'
implementation 'com.squareup.okhttp:okhttp-urlconnection:2.2.0'
implementation 'com.squareup.retrofit2:retrofit:2.3.0'
implementation 'com.squareup.retrofit2:converter-gson:2.3.0'
```

Add Below code in your launcher activity of app

```
import android.content.Intent;
    @Override
    public void onNewIntent(Intent intent) {
       this.setIntent(intent);
    }
```

Note : If you want Google Pay smooth payment, Then add below dependency and library in your project. Below section is mandatory to use payment through Google Pay Intent.

1.implementation **'com.google.android.gms:play-services-tasks:15.0.1'**
2. implementation files('**libs/gpay.aar**')

- To use google pay you need to add gpay.aar file into app/libs/ folder of the merchant application.

- Download gpay.aar file.

1.2. Change <Application> tag of androidManifest.xml file as below in your projects android folder.

android:allowBackup="true"

iOS Setup:

1. Copy easebuzz.framework of your application in embedded binaries.
2. Press + and add framework using 'Add other' button.
3. Browse framework: file from your folder and select 'copy items if needed'.
4. Set Always embed & sign from frameworks and Libraries and swift standard libraries to YES from project build settings

5. ***For React Native 0.60+***

1. npm install $(npm pack <Path of React Native SDK>/react-native-easebuzz-kit | tail -1)
Example : npm install $(npm pack HomeDirectory/SDKfolder/react-native-easebuzz-kit | tail -1)
(ignore if already installed)

2. cd ios && open podfile # Change the platform from iOS 9.0 to 10.0
pod install && cd .. # CocoaPods on iOS needs this extra step

For React Native 0.59 and lower

1. npm install $(npm pack <Path of React Native SDK>/react-native-easebuzz-kitt | tail -1)

Example : npm install $(npm pack HomeDirectory/SDKfolder/react-native-easebuzz-kit | tail -1)
(ignore if already installed)
2. Link the SDK with React Native Project using Xcode.
react-native link react-native-easebuzz-kit

6. Remove unused architectures, refer below link.
https://docs.easebuzz.in/mobile-integration-ios/remove-architectures

Your iOS setup is done.

React Native Integration Code (Java Script):

1. Write below JavaScript Code to Start Payment using Easebuzz Payment Gateway

1.1 Import following components

```
import {Platform, Button, NativeModules,NativeEventEmitter} from 'react-native';
import EasebuzzCheckout from 'react-native-easebuzz-kit';
```

1.2 Call Payment Method

Call EasebuzzCheckout.open method with the payment request parameters as options. This method returns a JS Promise where then part corresponds to a successful payment response or failure response and the catch part corresponds to any sdk failure i.e event failed etc.

```
const callPaymentGateway  = () => {
var options = {
   txnid: 'UNIQUE_TRANSACTION_ID',
   amount: 'Transaction amount in string as double format',
   productinfo: 'Product Information',
   firstname: 'Customer First Name',
   email: "customer@gmail.com",
   phone: "customer phone number",
   key: "YOUR_MERCHANT_KEY",
   udf1: "",
   udf2: "",
   udf3: "",
   udf4: "",
   udf5: "",
   s_url: "http://your.successurl.in",
   f_url: "http://your.failureurl.in",
   address1: "customer address 1",
   address2: "customer address 2",
```

```
    city: "customer city",
    state: "customer state",
    country: "customer country",
    zipcode: "customer zipcode",
    unique_id: "Customers unique ID only for save card feature otherwise pass empty",
    pay_mode: "This can be “test” or “production",
    hash: "Create hash using following procedure in hash generation section"
  }

EasebuzzCheckout.open(options).then((data) => {
    //handle the payment success & failed response here
    console.log("Payment Response:")
    console.log(data);
  }).catch((error) => {
    //handle sdk failure issue here
    console.log("SDK Error:")
    console.log(error);
  });
}
```

Note:

1. *For More description of request parameter and optional parameter refer below link.* https://docs.easebuzz.in/api/initiate-payment
2. *For More description of request and Response refer below link.* *https://docs.easebuzz.in/mobile-integration-react-native/handle-response*

Hash generation (sha512):

Hash is a mandatory parameter – used specifically to avoid any tampering during the transaction. It is sha512 encrypted string. And hash sequence is mentioned below.

Hash sequence:
key|txnid|amount|productinfo|firstname|email_id|udf1|udf2|udf3|udf4|udf5||||||salt|key

Generate the sha512 of above hash sequence. and pass as a hash parameter.

Note:

1. Make sure the parameters that you are passing to Easebuzz SDK intent should exactly be the same which has been used to generate the hash.

For example.

1. If you used demo@gmail.com to generate the hash and Demo@gmail.com is passed to SDK intent, Then It will throw an error.
2. If you appended space to any parameter while generating a hash and passed the space appended parameter to SDK intent then It will throw an error.
3. If you are using amount 1.00, Then It will throw an error. Please use amount like 1.0 (Complete number's amount). The amount like 1.12 will work fine.

Suggestion: It would be more secure if the hash generation process is done at back end (Server Side)