PayWithEaseBuzz Payment kit Integration (iOS)

iOS

The PaywithEaseBuzz iOS SDK makes it quick and easy to build an excellent payment experience in your iOS app. We provide powerful UI screens and elements that can be used out-of-the-box to collect your user's payment details. We also expose the low-level API's that power those UI's so that you can build fully custom experiences. See our Android Integration Guide to get started.

This SDK allows you to integrate payments via PaywithEaseBuzz into your native iOS app. It currently supports the following modes of payments:

1. Credit / Debit Cards
2. Netbanking
3. Wallets/Cash Cards
4. UPI
5. Ola-Money
6. EMI

Features

1. **Simplified Security:** With fraud detection and prevention mechanism, we provide the most protective layer for each transaction.We are also PCI-DSS compliant.
2. **Native UI:** We provide out-of-the-box native screens and elements so that you can get started quickly without having to think about designing the right interfaces.

Requirements

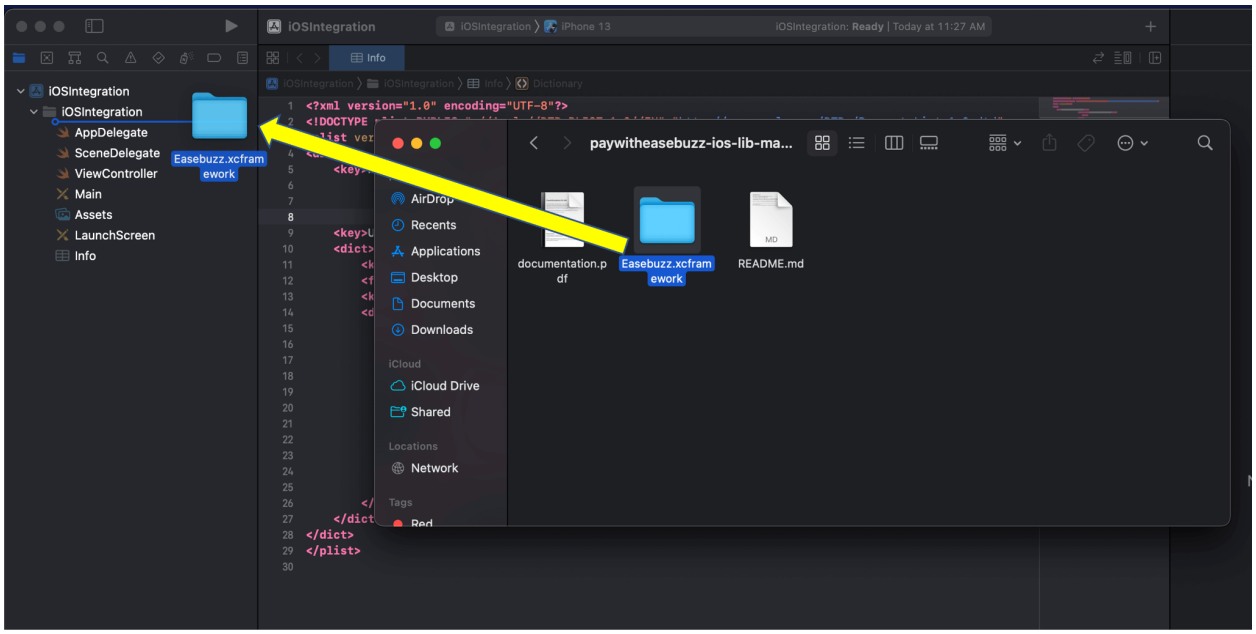
1. The PaywithEaseBuzz iOS SDK is compatible with apps supporting iOS 10 and above and requires Xcode 9.2 to build from source.

NOTE:

1. Certain card payments may not be supported on iOS 12.x and below due to a cookie management issue at certain payment switches. For a seamless experience, please use iOS 13.x and above.
2. According to PCI regulations, payment processing is not allowed on TLS v1 and TLS v1.1. Hence, if the device does not have TLS v1.2, the SDK will throw an error while initiating the payment . You can learn more about TLS versions here.

SDK Configuration

1. Copy Easebuzz.xcframework 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 swift standard libraries to YES from project build settings
   ALWAYS_EMBED_SWIFT_STANDARD_LIBRARIES = YES

5. To simply disable ATS, you can follow this steps by open Info.plist, and add the following lines:

```
<key>NSAppTransportSecurity</key>
    <dict> <key>NSAllowsArbitraryLoads</key>
      <true/>
    </dict>
```

6. To open upi psp app, you must make the following changes in your iOS app's Info.plist file.

```
<key>LSApplicationQueriesSchemes</key>
  <array>
  <string>tez</string>
  <string>phonepe</string>
  <string>paytm</string>
  <string>gpay</string>
  </array>
```

Initiate Payment

This is a mandatory and important step for initiating the payment. To generate an accesskey you have to integrate the Initiate Payment API at your backend. After you successfully call the Initiate Payment API then you will get an access key which is the value of the data key of the response. You can check the Initiate Payment API Doc Click here.

Note - It is mandatory to integrate the Initiate Payment API at your Backend only.

Integration code

After having successfully set up the sdk configuration and Initiate Payment API at your backend, write the below code to start payment on the click of Pay button from your app.

1. Import Easebuzz module in your ViewController

2. Set a Delegate to your ViewController as PayWithEasebuzzCallback and confirm the delegate.

3. On clicking the Pay button from your app, you need to call the initiatePaymentAction method. This function included one dictionary with access key and payment mode required parameters. And pass this dictionary to the Payment class. Before calling Easebuzz SDK, we need to generate an Access key using the Initiate Payment API.

   Swift Code -

```
import Easebuzz
    func initiatePaymentAction() {

        let orderDetails = [
        "access_key": "Pass access key generated in Initiate Payment API",
        "pay_mode": "This will either be “test” or “production"
        ] as [String:String]

        let payment = Payment.init(customerData: orderDetails)
        let paymentValid = payment.isValid().validity

        if !paymentValid {
            print("Invalid records")
        }else{
            PayWithEasebuzz.setUp(pebCallback: self )
            PayWithEasebuzz.invokePaymentOptionsView(paymentObj: payment,
isFrom: self)
        }
    }
```

Obj-C Code -

```
@import Easebuzz;
    @property (nonatomic,retain) Payment *payment;

    (void)initiatePaymentAction {
      NSDictionary *orderDetails =
      @{
        @"access_key": @"Pass access key generated in Initiate Payment API",
        @"pay_mode":@"This will either be “test” or “production"
      };

      _payment = [[Payment alloc]initWithCustomerData:orderDetails];

      BOOL paymentValid = _payment.isValid;
      if (!paymentValid) {
        NSLog(@"Invalid Print");
      }else{
        [PayWithEasebuzz setUpWithPebCallback:self];
        [PayWithEasebuzz invokePaymentOptionsViewWithPaymentObj:_payment
isFrom:self];
      }
    }
```

In the above sample code you have to pass the access_key and pay_mode.
The access key will be like -
“555a2b009214573bd833feca997244f1721ac69d7f2b09685911bc943dcf5201” and you will be get it from the initiate payment API. The **pay_mode** parameter will either be "test" or "production". Your key and salt will depend on the pay_mode which you pass.

Note:

1. Please do not change the name of the parameter from the orderDetails dictionary mentioned in the above code. The datatype of the value can be changed according to the value to be sent.

Handle Payment Response

You can receive the result and response in the callback delegate method.

Swift -

```
func PEBCallback(data: [String : AnyObject]) {

}
```

Obj-C -

```
- (void)PEBCallbackWithData:(NSDictionary * _Nonnull)data {
}
```

This result indicates the status of the transaction as success or failure. The detailed code is described below.

"payment_successfull"
"payment_failed"
"txn_session_timeout"
"back_pressed"
"user_cancelled"
"error_server_error"
"error_noretry"
"invalid_input_data"
"retry_fail_error"
"trxn_not_allowed"
"bank_back_pressed"

Your app can get the detailed response as follows :

Sample Code:

Swift -

```
func PEBCallback(data: [String : AnyObject]) {  // Handle Payment response_response key, It is a  anyObject
    let payment_response = data["payment_response"]
    print(payment_response ?? "")

    if payment_response as? [String:Any] != nil {
        // payment_response is Json Response
        print("Json response")
    }else{
        print("String response")
        //  payment_response is String
    }

     // Handle result Key : It should be in string
    let result = data["result"] as! String
}
```

Obj-C -

```
- (void)PEBCallbackWithData:(NSDictionary * _Nonnull)data {

    NSDictionary *responseDict = [data valueForKey:@"payment_response"];
    NSLog(@"%@",responseDict);

    // Handle Payment response_response key, It should be anyObject.
    BOOL isValid = [NSJSONSerialization isValidJSONObject:responseDict];
    if (isValid) {
     // payment_response is Json Response
    }else{
        //  payment_response is String
    }

    NSString *strResult = [data valueForKey:@"result"];
```

Response Description

1. Payment result values description and equivalent constants

Response	Value
StaticDataModel. TXN_SUCCESS_CODE	This a string constant and its value is “payment_successfull” result contains this value, if the payment transaction completed successfully.
StaticDataModel. TXN_TIMEOUT_CODE	PWEStaticDataModel.TXN_TIMEOUT_CODE is a string constant and its value is “txn_session_timeout” result contains this value, if the payment transaction failed because of the transaction time out.
StaticDataModel. TXN_BACKPRESSED_CODE	PWEStaticDataModel.TXN_BACKPRESSED_CODE is a string constant and its value is “back_pressed” result contains this value, if the user pressed the back button on coupons Activity.
StaticDataModel. TXN_USERCANCELLED_CODE	PWEStaticDataModel.TXN_USERCANCELLED_CODE is a string constant and its value is “user_cancelled” . The result contains this value, if the user pressed the cancel button during the payment process.
StaticDataModel. TXN_ERROR_SERVER_ERROR_CODE	PWEStaticDataModel.TXN_ERROR_SERVER_ERROR_CODE is a string constant and its value is “error_server_error” . The result contains this value, if the server side error occurred during the payment process.
StaticDataModel. TXN_ERROR_TXN_NOT_ALLOWED_CODE	PWEStaticDataModel.TXN_ERROR_TXN_NOT_ALLOWED_CODE is a string constant and its value is “trxn_not_allowed”
StaticDataModel.TXN_BANK_BACK_PRESSED_CODE	PWEStaticDataModel.TXN_BANK_BACK_PRESSED_CODE is a string constant and its value is “bank_back_pressed” result contains this value if user press the back button on bank page.
StaticDataModel. TXN_INVALID_INPUT_DATA_CODE	PWEStaticDataModel.TXN_INVALID_INPUT_DATA_CODE is a string constant and its value is “invalid_input_data’ result contains this value if payment request input parameters are not valid.
StaticDataModel. TXN_FAILED_CODE	PWEStaticDataModel.TXN_FAILED_CODE is a string constant and its value is “payment_failed” . result contains this value if payment fails from the bank side.
StaticDataModel. TXN_ERROR_NO_RETRY_CODE	PWEStaticDataModel.TXN_ERROR_NO_RETRY_CODE is a string constant and its value is “error_noretry”. This result can be considered as failed payment.
StaticDataModel. TXN_ERROR_RETRY_FAILED_CODE	PWEStaticDataModel.TXN_ERROR_NO_RETRY_CODE is a string constant and its value is “retry_fail_error”. This result can be considered as failed payment.

2. The below is a detailed response of payment

2.1 Success response json.


```
{
    "name_on_card": "Rizwan",
    "bank_ref_num": "214057035798",
    "udf3": "",
    "hash":
"5cd269f97088bf88ee9aef20864f6c86b738b4450851a7e518b21fc4bb2253ee153e6bd29ddc06a
adbc6a846d08b1452c488aed20f31c05ff6442894b994366a",
    "firstname": "Sagar",
    "net_amount_debit": "2.0",
    "payment_source": "Easebuzz",
    "surl": "http://localhost/paywitheasebuzz-php-lib-master/response.php",
    "error_Message": "Successful Transaction",
    "issuing_bank": "NA",
    "cardCategory": "NA",
    "phone": "8805596828",
    "easepayid": "E220520TZ7VGC2",
    "cardnum": "541919XXXXXX1220",
    "key": "X1X8XXX2XX",
    "udf8": "",
    "unmappedstatus": "NA",
    "PG_TYPE": "NA",
    "addedon": "2022-05-20 07:15:07",
    "cash_back_percentage": "50.0",
    "status": "success",
    "card_type": "Debit Card",
    "merchant_logo": "NA",
    "udf6": "",
    "udf10": "",
    "upi_va": "NA",
    "txnid": "EBZTXN0000357",
    "productinfo": "Test",
    "bank_name": "NA",
    "furl": "http://localhost/paywitheasebuzz-php-lib-master/response.php",
    "udf1": "",
    "amount": "2.0",
    "udf2": "",
    "udf5": "",
    "mode": "DC",
    "udf7": "",
    "error": "Successful Transaction",
    "udf9": "",
    "bankcode": "NA",
    "deduction_percentage": "0.0",
    "email": "sagar.sawant@easebuzz.in",
    "udf4": ""
}
```

2.2 Failure response json.


```
{
        "name_on_card": "Rizwan",
        "bank_ref_num": "",
        "udf3": "",
        "hash":
"bbf14e1a746abc6fba3267020d9d743c7952ad6c0927bacedac20b16eda0971ff734f98b15641805
5668a262f87b9e32b07476bca379992446359fd5bf1ce9d3",
        "firstname": "Sagar",
        "net_amount_debit": "20.0",
        "payment_source": "Easebuzz",
        "surl": "http://localhost/paywitheasebuzz-php-lib-master/response.php",
        "error_Message": "!ERROR!-GV00004-PARes status not sucessful.",
        "issuing_bank": "NA",
        "cardCategory": "NA",
        "phone": "8805596828",
        "easepayid": "E220520SIWNB81",
        "cardnum": "541919XXXXXX1220",
        "key": "D1K8SLB2XW",
        "udf8": "",
        "unmappedstatus": "NA",
        "PG_TYPE": "NA",
        "addedon": "2022-05-20 06:35:40",
        "cash_back_percentage": "50.0",
        "status": "failure",
        "card_type": "Debit Card",
        "merchant_logo": "NA",
        "udf6": "",
        "udf10": "",
        "upi_va": "NA",
        "txnid": "EBZTXN0000351",
        "productinfo": "Test",
        "bank_name": "NA",
        "furl": "http://localhost/paywitheasebuzz-php-lib-master/response.php",
        "udf1": "",
        "amount": "20.0",
        "udf2": "",
        "udf5": "",
        "mode": "DC",
        "udf7": "",
        "error": "!ERROR!-GV00004-PARes status not sucessful.",
        "udf9": "",
        "bankcode": "NA",
        "deduction_percentage": "0.0",
        "email": "sagar.sawant@easebuzz.in",
        "udf4": ""
 }
```