Pinelabs PHP SDK Integration

This document explains the integration process of the Pinelabs PHP SDK for your PHP applications

By Pinelabs Team

Installation Process

This section explains the installation flow of the PHP SDK for Pinelabs.

Prerequisites

Before installing the PHP library, make sure you meet the following prerequisites:

- PHP version 7.4 or higher, up to 8.1

Installation

If your project uses Composer for managing dependencies, you can easily install the PHP library by following these steps:

1. Open your terminal or command prompt.
2. Navigate to your project's root directory where your `**composer.json**` file is located.
3. Run the following Composer command to install the PHP library:

   “composer require pinelabs/php”

To locally add a PHP library to your composer.json file, follow these steps:

1. Download the PHP library and place it in a directory within your project. You can obtain the library files from a source like GitHub, or by downloading a release archive.
2. In your PHP library, make sure it has an autoloading mechanism defined, usually within a **composer.json** file. This is important to ensure that Composer can autoload the library's classes. An example **composer.json** file for the library might look like this:

```
{
    "name": "vendor-name/library-name",
    "autoload": {
        "psr-4": {
            "VendorName\\LibraryName\\": "src/"
        }
    }
}
```

3. Open your project's **composer.json** file and add a reference to the locally downloaded library. To do this, you can use the **path** repository type. Add a **repositories** section and specify the local path to the library, like this:

```
{
    "repositories": [
        {
            "type": "path",
            "url": "relative/path/to/library"
        }
    ],
    "require": {
        "vendor-name/library-name": "*"
    }
}
```

Example file using pinelab sdk

```
"repositories": [
        {
            "type": "path",
            "url": "./pinelab-sdk",
        }
    ],
    "require": {
        "pinelabs/php":"@dev"
    }
```

Replace "**relative/path/to/library**" with the actual path to the directory where the library is located within your project.

4. After updating your project's **composer.json** file, open your terminal or command prompt, navigate to your project's root directory (where the **composer.json** file is located), and run:

```
composer update
```

The composer will update the **composer.lock** file and autoload your locally added library.

Feature Provided by the SDK

The Pinelabs PHP library provides a set of features for integrating Pinelabs payment services into your application. Here are the supported features and some important information regarding its usage:

Supported Features

1. **Create Order**: You can use this feature to create orders for your customers, which can include product details, prices, and other relevant information required for payment processing.

2. **Fetch Order**: This feature allows you to retrieve information about existing orders. You can use it to check the status of an order or retrieve order details.

3. **EMI Calculator**: The library includes an EMI calculator, which can help your customers determine their Equated Monthly Installment (EMI) based on the payment amount, interest rate, and tenure.

4. **Hash Verification**: You can verify the integrity and authenticity of data received from Pinelabs using hash verification, ensuring the security of your transactions.

API Endpoint Information:

Here are the API endpoints for Pinelabs in both UAT (User Acceptance Testing) and production environments:

- **UAT (User Acceptance Testing) Environment**:
  Base URL: https://uat.pinepg.in/api/

- **Production Environment**:
  Base URL: https://pinepg.in/api/

Test Credentials

Here are the test merchant details that you can use for your Pinelabs integration in a PHP application. These details include the merchant ID, API access code, secret key, and a flag for test mode:

```
$merchantId = "106600";
$apiAccessCode = "bcf441be-411b-46a1-aa88-c6e852a7d68c";
$secret = "9A7282D0556544C59AFE8EC92F5C85F6";
$isTestMode = true;  // Set to false for production (default is false)
```

Code Implementation

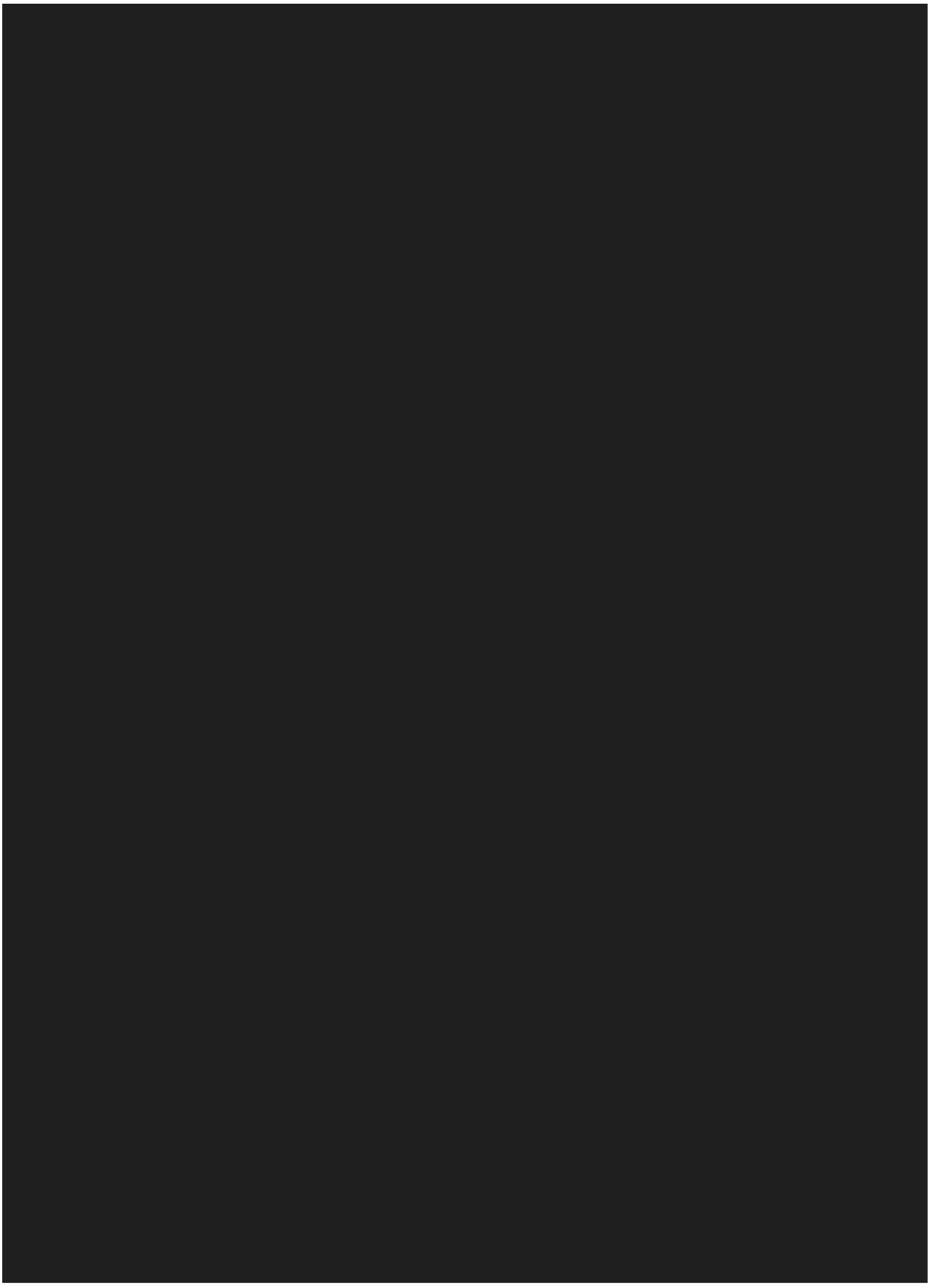
To create an API instance using the provided test merchant details, you can use the following code in your PHP application:

```
use \Pinelabs\Php\API;
$api = new API($merchantId, $apiAccessCode, $secret, $isTestMode);
```

1. Create Order API Payload

```
Txn (Order) Data ( Mandatory )
```php
$txn_data = [
    'txn_id' => 'orderId12345',  //Mandatory (unique id)
    'callback' => 'https://httpbin.org/post',  //Mandatory
    'amount_in_paisa' => "10000" // Mandatory  ( amount in paisa )
];
```

Customer Details (Optional)
```php
$customer_data = [
    'customer_id' =>'custId123', // Optional
    'first_name' => 'Ramsharan', // Optional
    'last_name' => 'Yadav', // Optional
    'email_id' => 'ramsharan@mcsam.in', // Optional
    'mobile_no' => '7737291210' // Optional
];
```
Billing Details (Optional)
```php
$billing_data = [
    'address1' => 'mcsam',  // Optional
    'address2' => 'mm tower',  // Optional
    'address3' => 'sector 18',  // Optional
    'pincode' => '122018',  // Optional
    'city' => 'Gurgaon',  // Optional
    'state' => 'Haryana',  // Optional
    'country' => 'India',  // Optional
];
```
Shipping Details (Optional)
```



```
// In these payment modes, the merchant can choose multiple modes which should be enabled on his merchant ID.
```

Product Details (Optional)

```php
$products_data = [
    [
        "product_code" => "testproduct02",  // Optional
        "product_amount" => 10000  // Optional
    ],
    .....
];
  // Accept multiple Array
```

Create Payment Link

Create a payment

```php
$response = $api->Payment()->Create($txn_data, $customer_data, $billing_data, $shipping_data, $udf_data, $payment_modes, $products_data);
echo '<pre>'; print_r($response); die;
```

Success Response

```
``` php
{"status":true,"token":"S01P3444qQM33XTlDJxg70bAj9II6AELtgU%2fNsFddGulYw%3d","redirect_url":"https://uat.pinepg.in/pinepg/v2/process/payment?token=S01P3444qQM33XTlDJxg70bAj9II6AELtgU%2fNsFddGulYw%3d"}
```
```

Failure Response

```
Array (
     [status] => false
     [message] => Something went wrong
     [respone_code] => -1
)
```

2. Fetch Order API Payload

Fetch Transaction details Request Body

```
Order Id ( Mandatory )
```php
$orderId = "orderId12345";

// The order ID which was sent by the user as unique transaction ID while
creating the order will be passed here.
```
```

```
Fetch a payment
```php
$response = $api->Payment()->Fetch($orderId);

echo '<pre>'; print_r($response); die;
```
```

Success Response

```
```php
{"ppc_MerchantID":"106600","ppc_MerchantAccessCode":"bcf441be-411b-46a1-aa
88-c6e852a7d68c","ppc_PinePGTxnStatus":"7","ppc_TransactionCompletionDateT
ime":"20\/09\/2023 04:07:52
PM","ppc_UniqueMerchantTxnID":"650acb67d3752","ppc_Amount":"1000","ppc_Txn
ResponseCode":"1","ppc_TxnResponseMessage":"SUCCESS","ppc_PinePGTransactio
nID":"12069839","ppc_CapturedAmount":"1000","ppc_RefundedAmount":"0","ppc_
AcquirerName":"BILLDESK","ppc_DIA_SECRET":"D640CFF0FCB8D42B74B1AFD19D97A37
5DAF174CCBE9555E40CC6236964928896","ppc_DIA_SECRET_TYPE":"SHA256","ppc_Pay
mentMode":"3","ppc_Parent_TxnStatus":"4","ppc_ParentTxnResponseCode":"1","
ppc_ParentTxnResponseMessage":"SUCCESS","ppc_CustomerMobile":"7737291210",
"ppc_UdfField1":"","ppc_UdfField2":"","ppc_UdfField3":"","ppc_UdfField4":"
","ppc_AcquirerResponseCode":"0300","ppc_AcquirerResponseMessage":"NA"}
```
```

Failure Response

```
{
      "Status":false,
      "message":"INVALID DATA",
      "respone_code":"-40"
}
```

```
IF Merchant Details Incorrect Then Return Response
```php
"IP Access Denied"
```
```

3. Fetch EMI Offers API

Request Body Payload

```
Txn (Order) Data ( Mandatory )
```php
$txn_data = [
    'amount_in_paisa' => "10000" // Mandatory  ( amount in paisa ) and sum
of product amount
];
```

Product Details (Optional)
```php
$products_data = [
    [
        "product_code" => "testproduct02",  // Mandatory
        "product_amount" => 10000  // Mandatory
    ]
];

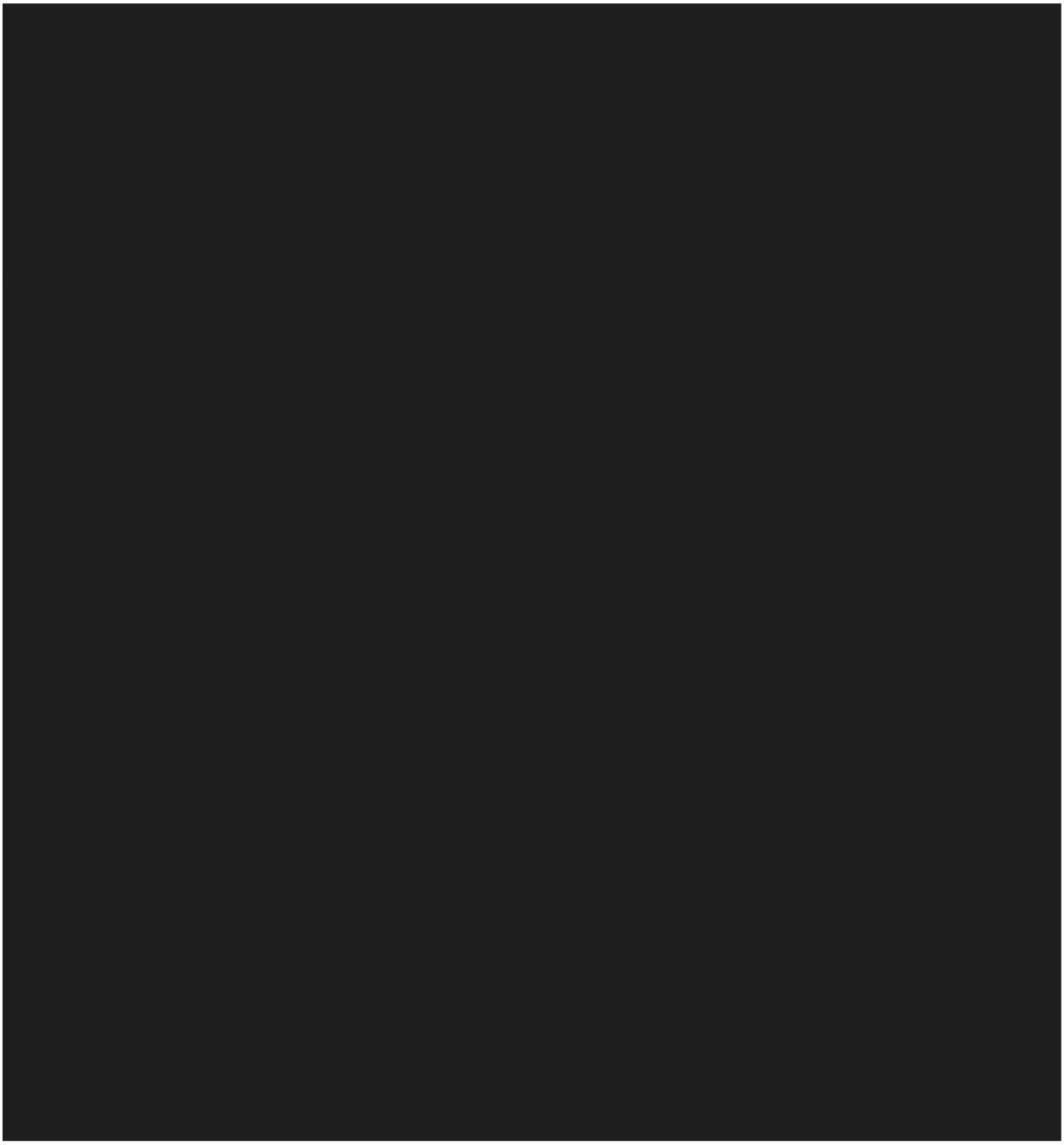
  // Accept only one Array
```
```

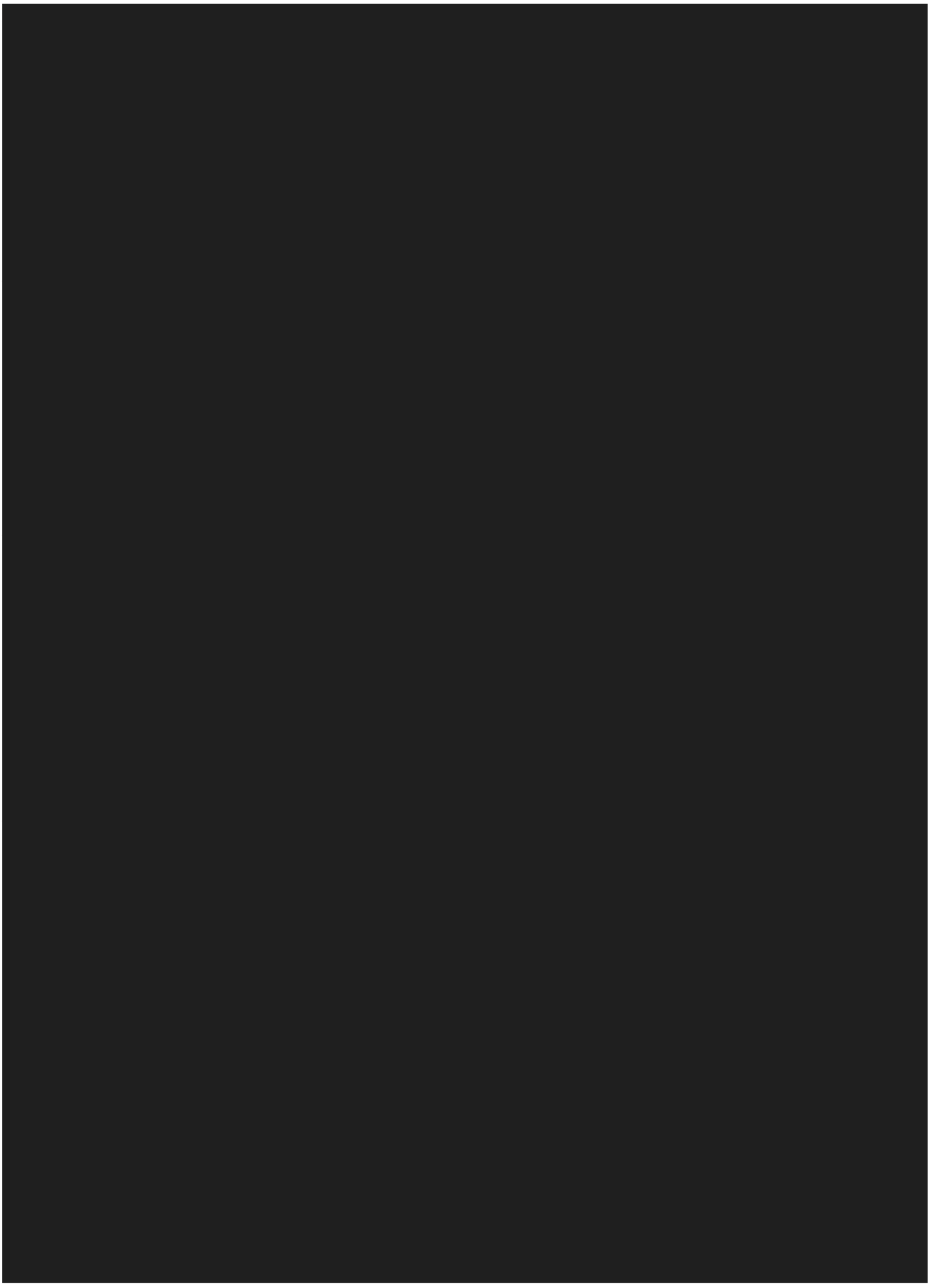
Call EMI Calculator

```
```php
$response = $api->EMI()->Calculator($txn_data, $products_data);

echo '<pre>'; print_r($response); die;
```
```

Success Response





```
:0,"bank_interest_rate_percentage":0,"bank_interest_rate":0}],"emi_scheme"
:{"scheme_id":48048,"program_type":105,"is_scheme_valid":true}},"tenure_id
":"96","tenure_in_month":"1","monthly_installment":0,"bank_interest_rate":
0,"interest_pay_to_bank":0,"total_offerred_discount_cashback_amount":0,"lo
an_amount":10000,"auth_amount":10000}],"issuer_name":"ICICI","is_debit_emi
_issuer":false},{"list_emi_tenure":[{"offer_scheme":{"product_details":[{"
schemes":[],"product_code":"testproduct02","product_amount":10000,"subvent
ion_cashback_discount":0,"product_discount":0,"subvention_cashback_discoun
t_percentage":0,"product_discount_percentage":0,"bank_interest_rate_percen
tage":0,"bank_interest_rate":0}],"emi_scheme":{"scheme_id":48043,"program_
type":105,"is_scheme_valid":true}},"tenure_id":"96","tenure_in_month":"1",
"monthly_installment":0,"bank_interest_rate":0,"interest_pay_to_bank":0,"t
otal_offerred_discount_cashback_amount":0,"loan_amount":10000,"auth_amount
":10000}],"issuer_name":"Kotak
Debit","is_debit_emi_issuer":true}],"response_code":1,"response_message":"
SUCCESS"}
```


Failure Response


```
{
    "status":false,
    "message":"Something went wrong",
    "respone_code":-40
}
```


4. Hash Verification API

Hash verification API required two Parameters

1. Received Hash
2. Callback Response

```
Hash ( Mandatory )
```php
$receviedHash = "475373549378937GJDFJGD8456834XCJBXJ4538VB67485";

// The hash received in response from Pinelabs.
```
```

```
Response Send In verify request for create a new hash ( Mandatory )
```

```php

//After removing the hash and the hash type from the response received from Pinelabs, we will send the response to the entity specified in the request hash.

// Sample Callback Response
$requestData = Array
(
    [merchant_id] => 106600
    [merchant_access_code] => bcf441be-411b-46a1-aa88-c6e852a7d68c
    [unique_merchant_txn_id] => 650c8d8ea61a0
    [pine_pg_txn_status] => 4
    [txn_completion_date_time] => 22/09/2023 12:08:29 AM
    [amount_in_paisa] => 10000
    [txn_response_code] => 1
    [txn_response_msg] => SUCCESS
    [acquirer_name] => BILLDESK
    [pine_pg_transaction_id] => 12072123
    [captured_amount_in_paisa] => 10000
    [refund_amount_in_paisa] => 0
    [payment_mode] => 3
    [mobile_no] => 7737291210
    [udf_field_1] =>
    [udf_field_2] =>
    [udf_field_3] =>
    [udf_field_4] =>
    [Acquirer_Response_Code] => 0300
    [Acquirer_Response_Message] => NA
    [parent_txn_status] =>
    [parent_txn_response_code] =>
    [parent_txn_response_message] =>
)
```

Code

```
Varify Hash
```php
```
```

```
$varify = $api->Hash()->Verify($receviedHash, $requestData);

echo $varify;
```

Success Response

```php
    true
```

Failure Response

```php
    false
```

Note:
Please note no additional charges like TDR, GST, etc are handled in our Plugins and the same needs to be manually handled at the merchant end.

TLS 1.2 information:
- PHP 5.5.19 and above should have TLS 1.2 support.
- For PHP 7 and above, TLS 1.2 support is typically available.