

SmartRoute Sample Code (php)

version 1.1

STS PayOne

27 April 2017

PROPRIETARY INFORMATION

STS PayOne and the STS PayOne logo are registered trademarks of STS Payment Network DMCC. Other trademarks and trade names may be used in this document to refer to either the entities claiming the marks and/or the names of their products. STS PayOne disclaims proprietary interest in the marks and names of others.

The information in this document has been reviewed and is believed to be accurate. However, neither STS PayOne nor its affiliates assume any responsibility for inaccuracies, errors, or omissions that may be contained herein. In no event will STS PayOne or its affiliates be liable for direct, indirect, special, incidental, or consequential damages resulting from any defect or omission in this document, even if advised of the possibility of such damages.

STS PayOne reserves the right to make improvements or changes to this document and information contained within, and to the products and services described at any time, without notice or obligation.

STS PayOne understands that any award to supply the products/ provide the services that are the subject of this document are subject to the mutual execution of a definitive written agreement.

All information supplied for the purpose of this document is to be considered STS PayOne confidential.

© 2016 STS PayOne

1 Contents

2 Introduction	4
2.1 Who should read this document?	4
2.2 Purpose	4
2.3 Prerequisites	4
3 Redirect Payment Message	5
3.1 Secure Hash Generation	5
3.2 Redirect Post Request Preparation	5
3.3 Redirect Post Request Submitting	6
3.4 Redirect Payment Response	7
4 Direct Post Payment Message	8
4.1 Secure Hash Generation:	8
4.2 Direct Post Request Preparation	9
4.3 Direct Post Request Submitting	9
4.4 Direct Post Payment Response	10
5 API MPayment	11
5.1 Secure Hash Generation	11
5.2 API MPayment Request	13
5.3 3DS Post Request Submitting to ACS URL	15
5.4 Sadad authentication Submission	16
6 API Payment	17
6.1 Secure Hash Generation	17
6.2 API Payment Request	17
6.3 Sadad authentication Submission	20
7 API Approve	20
7.1 Secure Hash Generation	20
7.2 API Approve Request	22
8 Inquiry Message	23
8.1 Secure Hash Generation	23
8.2 Inquiry Request	23
9 Refund Message	25
9.1 Secure Hash Generation	25
9.2 Refund Request	25

2 Introduction

2.1 Who should read this document?

This document is intended for merchant system designer and developer planning to integrate with SmartRoute interface to perform e-Commerce. Readers of this document should be web application developers.

2.2 Purpose

This guide is written for merchants who have signed up through SmartRoute system to use it as their Integration point with Payment Gateways for handling electronic transactions (payment, refund, confirm.. etc.) and from different payment methods (credit card, debit card...etc.), by using the HTTPS Post as programming interface to perform the transactions. In particular, it describes the format for sending transactions and the corresponding received responses.

2.3 Prerequisites

- Merchant is contracted with Payment Gateway as an e-Commerce Merchant.
- Merchant is contracted with SmartRoute as an Integration point with the Payment Gateways.
- Merchant is provided with a SmartRoute profile to generate Authentication Token used for integration.
- Merchant has a profile at Payment Gateway side.

3 Redirect Payment Message

- The following Extensions must be enabled in php: Mcrypt & Curl

3.1 Secure Hash Generation

```
1. $PAYONE_SECRET_KEY = "Y2FkMTdlOWZiMzJjMzY4ZGFkMzhkMWIz"; // Use Yours,
   Please Store Your Secret Key in safe Place(eg. database)
2. $transaction_id = (int)(microtime(true)*1000); //time in milleseconds
3.
4. $parameters = array();
5. // fill required parameters
6. $parameters['TransactionID'] = $transaction_id; //TMP:
7. $parameters['MerchantID'] = "ANBRedirectM";
8. $parameters['Amount'] = "2000";
9. $parameters['CurrencyISOCODE'] = "840";
10. $parameters['MessageID'] = "1";
11. $parameters['Quantity'] = "1";
12. $parameters['Channel'] = "0";
13. //fill some optional parameters
14. $parameters['Language'] = "en";
15. $parameters['ThemeID'] = "1000000001";
16. $parameters['ResponseBackURL'] = "https://MerchantSite/RedirectPaymentRespon
   sePage";// if this url is configured for the merchant it's not required
17. $parameters['Version'] = "1.0";
18.
19. //Create an Ordered String of The Parameters string with Secret Key by ksort
20. ksort($parameters);
21. $orderedString = $PAYONE_SECRET_KEY;
22. foreach($parameters as $k=>$param){
23.     $orderedString .= $param;
24. }
25. echo "orderdString: " . $orderedString.chr(10);
26.
27. // Generate SecureHash with SHA256
28. $secureHash = hash('sha256', $orderedString, false);
```

3.2 Redirect Post Request Preparation

```
1. // Step 2: Prepare Payment Request and Send It to Redirect PHP Page (To
   Send a Post Request)
2. $attributesData = array();
3.
4. $attributesData["TransactionID"] = $transaction_id;
5. $attributesData["MerchantID"] = "ANBRedirectM";
6. $attributesData["Amount"] = "2000";
7. $attributesData["CurrencyISOCODE"] = "840";
8. $attributesData["MessageID"] = "1";
9. $attributesData["Quantity"] = "1";
10. $attributesData["Channel"] = "0";
11. $attributesData["Language"] = "en";
12. $attributesData["ThemeID"] = "1000000001";
13. // if this url is configured for the merchant it's not required, else it is
   required
14. $attributesData["ResponseBackURL"] = "http://MerchantSite/RedirectPaymentRes
   ponsePage";
15. $attributesData["Version"] = "1.0";
16. $attributesData["RedirectURL"] = "http://SmartrouteURL/SmartRoutePaymentWEB/
   SRPayMsgHandler";
17. // set secure hash in the request
```

```

18.     $attributesData["SecureHash"] = $secureHash;
19.
20.     $_SESSION['SmartRouteParams'] = $attributesData;
21.     //redirect to "redirect.php";
22.     header('location: redirect.php');
23.     exit();

```

3.3 Redirect Post Request Submitting

```

1. <?php
2. if(!session_id()){
3.     session_start();
4. }
5.     // read the paramters from session
6.     $parameters      = $_SESSION['SmartRouteParams'];
7.     $redirectURL      = $parameters["RedirectURL"];
8.     $merchantID       = $parameters['MerchantID'];
9.     $amount           = $parameters['Amount'];
10.    $currencyCode      = $parameters['CurrencyISOCODE'];
11.    $language          = $parameters['Language'];
12.    $messageID         = $parameters['MessageID'];
13.    $transactionID     = $parameters['TransactionID'];
14.    $themeID           = $parameters['ThemeID'];
15.    $responseBackURL   = $parameters['ResponseBackURL'];
16.    $quantity          = $parameters['Quantity'];
17.    $channel           = $parameters['Channel'];
18.    $secureHash        = $parameters['SecureHash'];
19.    $version           = $parameters['Version'];
20.    ?>
21.    <html>
22.
23.
24.    <body onload="javascript:document.redirectForm.submit();">
25.    <form action="<?php echo $redirectURL?>" method="post" name="redirectForm">
26.    <input name="MerchantID" type="hidden" value="<?php echo $merchantID?>"/>
27.    <input name="Amount" type="hidden" value="<?php echo $amount?>"/>
28.    <input name="CurrencyISOCODE" type="hidden"
29.    value="<?php echo $currencyCode?>"/>
30.    <input name="Language" type="hidden" value="<?php echo $language?>"/>
31.    <input name="MessageID" type="hidden" value="<?php echo $messageID?>"/>
32.    <input name="TransactionID" type="hidden"
33.    value="<?php echo $transactionID?>"/>
34.    <input name="ThemeID" type="hidden" value="<?php echo $themeID?>"/>
35.    <input name="ResponseBackURL" type="hidden"
36.    value="<?php echo $responseBackURL?>"/>
37.    <input name="Quantity" type="hidden" value="<?php echo $quantity?>"/>
38.    <input name="Channel" type="hidden" value="<?php echo $channel?>"/>
39.    <input name="Version" type="hidden" value="<?php echo $version?>"/>
40.    <input name="SecureHash" type="hidden" value="<?php echo $secureHash?>"/>
41.    </form>
42.    </body>
43.    </html>

```

3.4 Redirect Payment Response

```
1. <?php
2.     $SECRET_KEY = "Y2FkMTdlOWZiMzJjMzY4ZGFkMzhkMWiz"; // Use Yours, Please Store
    Your Secret Key in safe Place(eg. database)
3.
4.     // get All Request Parameters
5.     $parameterNames = isset($_REQUEST)?array_keys($_REQUEST):[];
6.
7.     // store all response Parameters to generate Response Secure Hash
8.     // and get Parameters to use it later in your Code
9.     $responseParameters = [];
10.    foreach($parameterNames as $paramName){
11.        $responseParameters[$paramName] = filter_input(INPUT_REQUEST,$paramName)
12.    };
13.
14.    //order parameters by key using ksort
15.    ksort($responseParameters);
16.    $orderedString = $SECRET_KEY;
17.    foreach($responseParameters as $k=>$param){
18.        $orderedString .= $param;
19.    }
20.
21.    echo "--- Ordered String ---".chr(10);
22.    echo $orderedString.chr(10);
23.
24.    // Generate SecureHash with SHA256
25.    $secureHash = hash('sha256', $orderedString, false);
26.
27.    // get the received secure hash from result map
28.    $receivedSecureHash = filter_input(INPUT_REQUEST,'Response.SecureHash');
29.
30.    // Now that we have the map, order it to generate secure hash and compare it
    with the received one
31.    if($receivedSecureHash !== $secureHash){
32.        // IF they are not equal then the response shall not be accepted
33.        echo "Received Secure Hash does not Equal generated Secure hash";
34.    }else{
35.        // Complete the Action get other parameters from result map and do
36.        // your processes
37.        // Please refer to The Integration Manual to see the List of The
38.        // Received Parameters
39.        echo "Status is: ".filter_input(INPUT_REQUEST,'Response.Status');
40.    }
```

4 Direct Post Payment Message

4.1 Secure Hash Generation:

```
1. <?php
2.     $SECRET_KEY = "Y2FkMTdlOWZiMzJjMzY4ZGFkMzhkMWIz"; // Use Yours, Please Store
    Your Secret Key in safe Place(eg. database)
3.
4.     // get All Request Parameters
5.     $parameterNames = isset($_REQUEST)?array keys($_REQUEST):[];
6.
7.     // store all response Parameters to generate Response Secure Hash
8.     // and get Parameters to use it later in your Code
9.     $responseParameters = [];
10.    foreach($parameterNames as $paramName){
11.        $responseParameters[$paramName] = filter input(INPUT_REQUEST,$paramName)
12.    };
13.
14.    //order parameters by key using ksort
15.    ksort($responseParameters);
16.    $orderedString = $SECRET KEY;
17.    foreach($responseParameters as $k=>$param){
18.        $orderedString .= $param;
19.    }
20.
21.    echo "--- Ordered String ---".chr(10);
22.    echo $orderedString.chr(10);
23.
24.    // Generate SecureHash with SHA256
25.    $secureHash = hash('sha256', $orderedString, false);
26.
27.    // get the received secure hash from result map
28.    $receivedSecureHash = filter input(INPUT_REQUEST,'Response.SecureHash');
29.
30.    // Now that we have the map, order it to generate secure hash and compare it
    with the received one
31.    if($receivedSecureHash !== $secureHash){
32.        // IF they are not equal then the response shall not be accepted
33.        echo "Received Secure Hash does not Equal generated Secure hash";
34.    }else{
35.        // Complete the Action get other parameters from result map and do
36.        // your processes
37.        // Please refer to The Integration Manual to see the List of The
38.        // Received Parameters
39.        echo "Status is: ".filter input(INPUT_REQUEST,'Response.Status');
40.    }
```

4.2 Direct Post Request Preparation

```
1. // Step 2: Prepare Payment Request and Send It to Redirect Page (To Send a Post Request)
2.     $paymentParameters = [];
3.     $paymentParameters["TransactionID"] = $transactionId;
4.     $paymentParameters["MerchantID"] = "ANBRedirectM";
5.     $paymentParameters["Amount"] = "2000";
6.     $paymentParameters["CurrencyISOCODE"] = "840";
7.     $paymentParameters["MessageID"] = "1";
8.     $paymentParameters["Quantity"] = "1";
9.     $paymentParameters["Channel"] = "0";
10.
11.     $paymentParameters["PaymentMethod"] = "1";
12.
13.     // $paymentParameters["SadadOlpId"] = "testSadad";
14.     $paymentParameters["Language"] = "en";
15.     $paymentParameters["ThemeID"] = "1000000001";
16.     $paymentParameters["ResponseBackURL"] =
17.     "https://MerchantSite/RedirectPaymentResponsePage"; // if this url is
    configured for the merchant it's not required
18.     $paymentParameters["Version"] = "1.0";
19.     $paymentParameters["RedirectURL"] = "http://localhost:9080/SmartRoutePayment
    WEB/SRPayMsgHandler";
20.
21.     // set secure hash in the request
22.     $paymentParameters["SecureHash"] = $secureHash;
23.
24.     $_SESSION['PaymentParams'] = $paymentParameters;
25.     header('location: submitRedirectPayment.php');
```

4.3 Direct Post Request Submitting

```
1. <?php
2. if(!session_id()){
3.     session_start();
4. }
5. ?>
6. <!-- STEP 3: Create PHP Page send Request -->
7. <?php
8. // read the parameters from request
9.     $paymentParameters = $_SESSION["PaymentParams"];
10.     $redirectURL = (String) $paymentParameters["RedirectURL"];
11.     $amount = (String) $paymentParameters["Amount"];
12.     $currencyCode = (String) $paymentParameters["CurrencyISOCODE"];
13.     $transactionID = (String) $paymentParameters["TransactionID"];
14.     $merchantID = (String) $paymentParameters["MerchantID"];
15.     $language = (String) $paymentParameters["Language"];
16.     $messageID = (String) $paymentParameters["MessageID"];
17.     $secureHash = (String) $paymentParameters["SecureHash"];
18.     $themeID = (String) $paymentParameters["ThemeID"];
19.     $responseBackURL = (String) $paymentParameters["ResponseBackURL"];
20.     $channel = (String) $paymentParameters["Channel"];
21.     $quantity = (String) $paymentParameters["Quantity"];
```

```

22.     $version = (String) $paymentParameters["Version"];
23.
24.     $paymentMethod = (String) $paymentParameters["PaymentMethod"];
25.     //optional for sadad
26.     $sadadOlpId = (String) $paymentParameters['SadadOlpId'];
27.     ?>
28.
29.     <html>
30.     <body>
31.     <form action="<?php echo $redirectURL?>" method="post" name="redirectForm">
32.     <input name="MerchantID" type="hidden" value="<?php echo $merchantID?>" />
33.     <input name="Amount" type="hidden" value="<?php echo $amount?>" />
34.     <input name="CurrencyISOCODE" type="hidden"
value="<?php echo $currencyCode?>" />
35.     <input name="Language" type="hidden" value="<?php echo $language?>" />
36.     <input name="MessageID" type="hidden" value="<?php echo $messageID?>" />
37.     <input name="TransactionID" type="hidden"
value="<?php echo $transactionID?>" />
38.     <input name="ThemeID" type="hidden" value="<?php echo $themeID?>" />
39.     <input name="ResponseBackURL" type="hidden"
value="<?php echo $responseBackURL?>" />
40.     <input name="Quantity" type="hidden" value="<?php echo $quantity?>" />
41.     <input name="Channel" type="hidden" value="<?php echo $channel?>" />
42.     <input name="Version" type="hidden" value="<?php echo $version?>" />
43.     <input name="PaymentMethod" type="hidden"
value="<?php echo $paymentMethod?>" />
44.     <input name="SadadOlpId" type="hidden" value="<?php echo $sadadOlpId ?>" />
45.     <label>Card Number</label>
46.     <input name="CardNumber" type="text" value=""/>
47.     <br/>
48.     <label>Card Holder Name</label>
49.     <input name="CardHolderName" type="text" value=""/>
50.     <br/>
51.     <label>Security Code</label>
52.     <input name="SecurityCode" type="text" value=""/>
53.     <br/>
54.     <label>Year Expiry Date</label>
55.     <input name="ExpiryDateYear" type="text" value=""/>
56.     <br/>
57.     <label>Month Expiry Date</label>
58.     <input name="ExpiryDateMonth" type="text" value=""/>
59.     <br/>
60.     <input name="SecureHash" type="hidden" value="<?php echo $secureHash ?>" />
61.     <input type="submit" value="Proceed" />
62.     </form>
63.     </body>
64.     </html>

```

4.4 Direct Post Payment Response

```

1. <?php
2. //4.4 Direct Post Payment Response
3.

```

```

4. $SECRET_KEY = "Y2FkMTdlOWZiMzJjMzY4ZGFkMzhkMWIz";// Use Yours, Please Store Your
   Secret Key in safe Place (eg. database)
5. // get All Request Parameters
6. $parameters = $_REQUEST;
7. // store all response Parameters to generate Response Secure Hash, sort them
8. // and get Parameters to use it later in your Code
9. ksort($parameters);
10. // Now that we have the map, order it to generate secure hash and compare it
   with the received one
11. $responseOrderdString = "";
12. $responseOrderdString .= $SECRET_KEY;
13. $responseOrderdString .= implode(' ', $parameters);
14.
15. echo ("Response Orderd String is: " . $responseOrderdString).chr(10);
16.
17. // Generate SecureHash with SHA256
18. $generatedsecureHash = hash('sha256', $responseOrderdString, false);
19. // get the received secure hash from result map
20. $receivedSecurehash = $parameters["Response.SecureHash"];
21. if ($receivedSecurehash !== $generatedsecureHash) {
22.     // IF they are not equal then the response shall not be accepted
23.     echo "Received Secure Hash does not Equal generated Secure hash";
24. } else {
25.     // Complete the Action get other parameters from result map and do
26.     // your processes
27.     // Please refer to The Integration Manual to See The List of The
28.     // Received Parameters
29.     $status = $parameters["Response.Status"];
30.     echo "Status is: " . $status;
31. }

```

5 API MPayment

5.1 Secure Hash Generation

```

1. <?php
2. //Step 1: Generate Secure Hash
3. $SECRET_KEY = "Y2FkMTdlOWZiMzJjMzY4ZGFkMzhkMWIz"; // Use Yours, Please Store
   Your Secret Key in safe Place(eg. database)
4.
5. // put the parameters in a array to have the parameters sorted
   alphabetically later via ksort.
6. $parameters = array();
7.
8. $transactionId = (int)(microtime(true)*1000); //time in milleseconds
9.
10. // fill required parameters
11. $parameters["TransactionID"] = $transactionId;
12. $parameters["MerchantID"] = "ANBRedirectM";
13. $parameters["Amount"] = "2000";
14. $parameters["CurrencyISOCODE"] = "840";
15. $parameters["MessageID"] = "8";
16. $parameters["Quantity"] = "1";
17. $parameters["Channel"] = "0";
18. $parameters["PaymentMethod"] = "1";

```

```
19.     $parameters["ClientIpAddress"] = "127.0.0.1";
20.
21.     //for Card Payment (conditional;paymentMethod=1)
22.     $parameters["CardNumber"] = "4012001045873335";
23.     $parameters["ExpiryDateYear"] = "01";
24.     $parameters["ExpiryDateMonth"] = "19";
25.     $parameters["SecurityCode"] = "123";
26.     $parameters["CardHolderName"] = "1";
27.
28.     //for Sadad Payment (conditional;paymentMethod=2)
29.     //$parameters["SadadOlpId"] = "testSadad";
30.     //$parameters["mfu"] = "https://MerchantSite/RedirectPaymentRequestPage";
31.
32.     //fill some optional parameters
33.     $parameters["Language"] = "en";
34.     $parameters["ThemeID"] = "1000000001";
35.     $parameters["Version"] = "1.0";
36.
37.
38.     //Create an Ordered String of The Parameters Map with Secret Key
39.     ksort($parameters);
40.     $orderedString = $SECRET_KEY;
41.     foreach($parameters as $k=>$param){
42.         $orderedString .= $param;
43.     }
44.
45.     echo "Ordered String: ".chr(10).$orderedString.chr(10);
46.
47.     // Generate SecureHash with SHA256
48.     $secureHash = hash('sha256', $orderedString, false);
49.
```

5.2 API MPayment Request

```
1. // if the Card was 3DS Enrolled, APIPayment Will be Divided into two requests.
2. //in the response, if the received status code was "20001" or "20002" this means
3. //that the Payment is 3DS supported, which means you need to authenticate with the
4. //Bank site, all needed parameters for 3DS in will be included in the response,
5. //and after Authentication, you will send APIApprove Request to SmartRoute.
6. //Note: The Difference between 3DS payment and none-3DS Payment, will start after
   getting the APIPayment response.
7.
8. $queryStringArr = [
9.     "TransactionID" => $transactionId,
10.    "MerchantID" => "ANBRedirectM",
11.    "Amount" => "2000",
12.    "CurrencyISOCode" => "840",
13.    "MessageID" => "8",
14.    "Quantity" => "1",
15.    "Channel" => "0",
16.    "PaymentMethod" => "1",
17.    "ClientIPAddress" => "127.0.0.1",
18.    //for Card Payment (conditional.append("&")paymentMethod=1)
19.    "CardNumber" => "4012001045873335",
20.    "ExpiryDateYear" => "01",
21.    "ExpiryDateMonth" => "19",
22.    "SecurityCode" => "123",
23.    "CardHolderName" => "1",
24.    //for Sadad Payment (conditional.append("&")paymentMethod=2)
25.    "SadadOlpId" => "testSadad",
26.    "mfu" => "https://MerchantSite/RedirectPaymentRequestPage",
27.    //fill some optional parameters
28.    "Language" => "en",
29.    "ThemeID" => "1000000001",
30.    "Version" => "1.0",
31.    "SecureHash" => $secureHash,
32. ];
33.
34. //Send the request
35. $newRequestQuery = http_build_query($queryStringArr);
36.
37. $url = "https://SR_URL";
38. $ch = curl_init($url);
39. curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
40. //write parameters
41. curl_setopt($ch, CURLOPT_POST, true);
42. curl_setopt($ch, CURLOPT_POSTFIELDS, $newRequestQuery);
43.
44. // Get the response
45. $output = curl_exec($ch);
46. curl_close($ch);
47. //Output the response
48. echo $output;
49.
50. // To read the output string you might want to split it
51. // on '&' to get pairs then on '=' to get name and value
52. // and for a better and ease on verifying secure hash you should put
53. $result = [];
54. parse_str($output, $result);
55. ksort($result);
56.
57. // Now that we have the map, order it to generate secure hash and compare it
   with the received one
58. $responseOrderdString = $SECRET_KEY;
```

```

59.     foreach($result as $res_k=>$result_v){
60.         $responseOrderdString .= $result_v;
61.     }
62.
63.     echo "-- Response Orderd String --".chr(10);
64.     echo $responseOrderdString.chr(10);
65.
66.     // Generate SecureHash with SHA256
67.     $generatedsecureHash = hash('sha256',$responseOrderdString);
68.
69.     // get the received secure hash from result map
70.     $receivedSecurehash = $result['Response.SecureHash'];
71.
72.     if($receivedSecurehash == $generatedsecureHash){
73.         // IF they are not equal then the response shall not be accepted
74.         echo "Received Secure Hash does not Equal generated Secure hash".chr(10);
75.     }else{
76.
77.         // Complete the Action get other parameters from result map and do
78.         // your processes
79.         // Please refer to The Integration Manual to See The List of The
80.         // Received Parameters
81.         $status = $result["Response.Status"];
82.         echo "Status is :" . $status.chr(10);
83.         if ("20001" === $status) {
84.             // if the received status code was 20001 this means that this
transaction needs 3DS
85.             //Authentication, the parameters you need are received with
the response too.
86.             // prepare parameters to send to JSP , to Send it to 3DS in A
Post Request
87.             $bankUrl = (String)$result["Response.AcsURL"];
88.             $PaRequestMessage = (String)$result["Response.PaRequestMessage
"];
89.             // 3DS Response page ( the url that you want 3DS
Authentication to forward the request to)
90.             $Merchant3DSResponseURL= "http://yoursite/your3DSResponsepage"
;
91.             $_SESSION["ACSURL"] = $bankUrl;
92.             $_SESSION["3DSPaMessage"] = $PaRequestMessage;
93.             $_SESSION["TERMURL_PREFIX"] = $Merchant3DSResponseURL;
94.
95.             // Verification Enrollment Result Used for 3DS payment.
96.             $veResult= (String)$result["Response.ResponseVeResult"];
97.
98.             /*****
99.             /*****
100.            /*****
101.            /*STORE veResult IN DATABASE OR ANY SAFE PLACE TO USE IT IN
APPROVE REQUEST*/
102.            /*****
103.            /*****
104.            /*****
105.            header("location: RedirectTo3DS.php");
106.            exit();
107.        }else if("20002" === $status) {
108.            // this means that the transaction needs Sadad Authentication
109.            $responseEstn = $result["Response.estn"];
110.            $responseMfu = $result["Response.mfu"];
111.            $responseAuthenticationUrl = $result["Response.AuthenticationU
RL"];
112.            $_SESSION["responseEstn"] = $responseEstn;
113.            $_SESSION["responseMfu"] = $responseMfu;
114.            $_SESSION["responseAuthenticationUrl"] = $responseAuthenticati
onUrl;

```

```

115.
116.         header("location: AuthenticateSadad.php");
117.         exit();
118.     }else {
119.         // then the card is not 3ds enrolled
120.         // this means your payment has been completed
121.         echo "Status is :". $status;
122.     }
123. }

```

5.3 3DS Post Request Submitting to ACS URL

```

1. <?php
2. ob_start();
3. if(!session_id()){
4.     session_start();
5. }
6. //get the error code message
7. $ACSURL="";
8. $PaMessage="";
9. $TermsURL = "";
10.     try{ //set the labels
11.         $ACSURL = (String)$_SESSION["ACSURL"];
12.         if ($ACSURL == null) { //the errorCode is null
13.             $ACSURL="";
14.         } else { //there is an error occurred
15.             $PaMessage = (String)$_SESSION["3DSPaMessage"];
16.         }
17.     }catch(Exception $e){
18.         echo ("Exception:". $e->getMessage());
19.     }
20.     ?>
21.     <html>
22.     <body onload="javascript:document.ACSAutoSubmitForm.submit();">
23.     <form name="ACSAutoSubmitForm" action="<?php $ACSURL ?>" method="POST">
24.     <input type="hidden" name="PaReq" value="<?php $PaMessage ?>">
25.     <input type="hidden" name="TermUrl"
26.     value="<?php urlencode($_SESSION["TERMURL_PREFIX"]) ?>">
27.     <input type="hidden" name="MD" value="testing">
28.     </form>
29.     </body>
30.     </html>

```

5.4 Sadad authentication Submission

```
1. <?php
2. ob_start();
3. if(!session_id()){
4.     session_start();
5. }
6. $responseEstn = (String) $_SESSION["responseEstn"];
7. $responseMfu = (String) $_SESSION["responseMfu"];
8. $ibUrl = (String) $_SESSION["responseAuthenticationUrl"];
9. ?>
10.
11.
12.     <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
13.
14.     <html language="php" contentType="text/html; charset=windows-1256">
15.     <head>
16.     <meta http-equiv="Content-Type" content="text/html; charset=windows-1256">
17.     <meta http-equiv="Pragma" content="no-cache">
18.     <meta http-equiv="Expires" content="-1">
19.     <title>3DS Authentication</title>
20.     <script>
21.     function redirectForm() {
22.     document.AutoSubmitForm.submit();
23.     }
24.     </script>
25.     </head>
26.     <body>
27.     <form name="AutoSubmitForm" action="<?php echo $ibUrl ?>" method="GET">
28.     <input type="hidden" name="mfu" value="<?php echo $responseMfu ?>">
29.     <input type="hidden" name="estn" value="<?php echo $responseEstn ?>">
30.     </form>
31.     <input name="submit" value="Authenticate" type="button"
32.     onclick="redirectForm()" >
33.     </body>
34.     </html>
```

6 API Payment

6.1 Secure Hash Generation

```
1. //Step 1: Generate Secure Hash
2. $SECRET_KEY = "Y2FkMTdlOWZiMzJjMzY4ZGFkMzhkMWiz"; // Use Yours, Please Store
   Your Secret Key in safe Place(e.g. database)
3. // put the parameters in a array to have the parameters to have them sorted
   alphabetically via ksort.
4. $parameters = [];
5. $transactionId = (int)microtime(true)*1000; //output to be like: 1495004320389
6. // fill required parameters
7. $parameters["TransactionID"] = $transactionId;
8. $parameters["MerchantID"] = "ANBRedirectM";
9. $parameters["Amount"] = "2000";
10. $parameters["CurrencyISOCODE"] = "840";
11. $parameters["MessageID"] = "9";
12. $parameters["Quantity"] = "1";
13. $parameters["Channel"] = "0";
14. $parameters["PaymentMethod"] = "1";
15. $parameters["ClientIPAddress"] = "127.0.0.1";
16. //for Card Payment (conditional;paymentMethod=1)
17. $parameters["CardNumber"] = "4012001045873335";
18. $parameters["ExpiryDateYear"] = "01";
19. $parameters["ExpiryDateMonth"] = "19";
20. $parameters["SecurityCode"] = "123";
21. $parameters["CardHolderName"] = "1";
22. //for Sadad Payment (conditional;paymentMethod=2)
23. // $parameters["SadadOlpId"] = "testSadad";
24. // $parameters["mfu"] = "https://MerchantSite/RedirectPaymentRequestPage";
25.
26. //fill some optional parameters
27. $parameters["Language"] = "en";
28. $parameters["ThemeID"] = "1000000001";
29. $parameters["Version"] = "1.0";
30. //Create an Ordered String of The Parameters Map with Secret Key
31.
32. ksort($parameters);
33. $orderedString = $SECRET_KEY;
34. foreach($parameters as $param){
35.     $orderedString .= $param;
36. }
37. echo ("orderdString ". $orderedString);
38. // Generate SecureHash with SHA256
39. // Using DigestUtils from apache.commons.codes.jar Library
40. $secureHash = hash('sha256', $orderedString, false);
41.
```

6.2 API Payment Request

```
1. //in the response, if the received status code was "20002" it needs Sadad
   authentication,
2. //and after Authentication, you will send APIApprove Request to SmartRoute.
3. $requestQueryArr = [
4.     "TransactionID" => $transactionId,
5.     "MerchantID" => "ANBRedirectM",
6.     "Amount" => "2000",
7.     "CurrencyISOCODE" => "840",
8.     "MessageID" => "9",
9.     "Quantity" => "1",
```

```

10.         "Channel" => "0",
11.         "PaymentMethod" => "1",
12.         "ClientIpAddress" => "127.0.0.1",
13.         //for Card Payment (conditional.append("&")paymentMethod=1)
14.         "CardNumber" => "4012001045873335",
15.         "ExpiryDateYear" => "01",
16.         "ExpiryDateMonth" => "19",
17.         "SecurityCode" => "123",
18.         "CardHolderName" => "1",
19.         "SecureHash" => $secureHash,
20.
21.         //for Sadad Payment (conditional.append("&")paymentMethod=2)
22.         "SadadOlpId" => "testSadad",
23.         "mfu" => "https://MerchantSite/RedirectPaymentRequestPage",
24.
25.         //fill some optional parameters
26.         "Language" => "en",
27.         "ThemeID" => "1000000001",
28.         "Version" => "1.0",
29.         "SecureHash" => $secureHash,
30.     ];
31.     //configure query string
32.     $newRequestQuery = http build query($requestQueryArr);
33.
34.     //Send the request
35.     $ch = curl_init("https://SR_URL");
36.     curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
37.     curl_setopt($ch, CURLOPT_POST, true);
38.     curl_setopt($ch, CURLOPT_POSTFIELDS, $newRequestQuery);
39.
40.     // Get the response
41.     $output = curl_exec($ch);
42.     curl_close($ch);
43.
44.     //Output the response
45.     echo ($output);
46.     // this string is formatted as a "Query String" -
    name=value&name2=value2.....
47.
48.     // To read the output string you might want to split it
49.     // on '&' to get pairs then on '=' to get name and value using parse_str then
    order it using ksort
50.     $result = [];
51.     parse_str($output, $result);
52.     ksort($result);
53.
54.     // Now that we have the map, order it to generate secure hash and compare it
    with the received one
55.     $responseOrderdString = $SECRET_KEY;
56.     foreach($result as $resParam) {
57.         $responseOrderdString .= $resParam;
58.     }
59.     echo "Response Orderd String is " . $responseOrderdString;
60.
61.     // Generate SecureHash with SHA256
62.     $generatedsecureHash = hash('sha256', $responseOrderdString);
63.
64.     // get the received secure hash from result map
65.     $receivedSecurehash = $result['Response.SecureHash'];
66.
67.     if($receivedSecurehash !== $generatedsecureHash){
68.         //IF they are not equal then the response shall not be accepted
69.         echo ("Received Secure Hash does not Equal generated Secure hash");
70.     }else{

```

```

71.          // complete the Action get other parameters from result map and do
           your processes
72.          // please refer to The Integration Manual to See The List of The
           Received Parameters
73.          $status= $result["Response.Status"];
74.          echo ("Status is :". $status);
75.          if("20002" == $status) {
76.              $responseEstn = $result["Response.estn"];
77.              $responseMfu = $result["Response.mfu"];
78.              $responseAuthenticationUrl = $result["Response.AuthenticationU
RL"];
79.              $_SESSION["responseEstn"] = $responseEstn;
80.              $_SESSION["responseMfu"] = $responseMfu;
81.              $_SESSION["responseAuthenticationUrl"] = $responseAuthenticati
onUrl;
82.              header("location: AuthenticateSadad.php");
83.              exit();
84.          }else {
85.              // then the card is not 3ds enrolled
86.              // this means your payment has been completed
87.              echo ("Status is :". $status);
88.          }
89.      }
90.

```

6.3 Sadad authentication Submission

```
1. <?php
2. ob_start();
3. if(!session_id()){
4.     session_start();
5. }
6. $responseEstn = (String) $_SESSION["responseEstn"];
7. $responseMfu = (String) $_SESSION["responseMfu"];
8. $ibUrl = (String) $_SESSION["responseAuthenticationUrl"];
9. ?>
10. <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
11. <html language="php" contentType="text/html; charset=windows-1256">
12. <head>
13. <meta http-equiv="Content-Type" content="text/html; charset=windows-1256">
14. <meta http-equiv="Pragma" content="no-cache">
15. <meta http-equiv="Expires" content="-1">
16. <title>3DS Authentication</title>
17. <script>
18. function redirectForm() {
19. document.AutoSubmitForm.submit();
20. }
21. </script>
22. </head>
23. <body>
24. <form name="AutoSubmitForm" action="<?php echo $ibUrl ?>" method="GET">
25. <input type="hidden" name="mfu" value="<?php echo $responseMfu ?>">
26. <input type="hidden" name="estn" value="<?php echo $responseEstn ?>">
27. </form>
28. <input name="submit" value="Authenticate" type="button"
   onclick="redirectForm()" >
29. </body>
30. </html>
```

7 API Approve

7.1 Secure Hash Generation

```
1. <?php
2. //Step 1: Generate Secure Hash
3. $SECRET_KEY = "Y2FkMTdlOWZiMzJjMzY4ZGFkMzhkMWIz"; // Use Yours, Please Store
   Your Secret Key in safe Place(e.g. database)
4. // put the parameters in a TreeMap to have the parameters to have them sorted
   alphabetically.
5. $parameters = [];
6. $transactionId = (int)microtime(true)*1000; //output to be like: 1495004320389
7.
8. // fill required parameters
9. $parameters["MessageID"] = "1";
10. $parameters["TransactionID"] = $transactionId;
11. $parameters["MerchantID"] = "ANBRedirectM";
12. $parameters["PaymentMethod"] = "1";
13.
```

```

14. //it's real length is very longer than this this value is just a sample
15. $parameters["PAREs"] = urlencode("eJzFV2mTokoW/Ssd9T4a3ewiHZQvkh2");
16.
17. //in case of sadad payment
18. //$_SESSION["estn"] = "Test123";
19.
20. //for Card Payment (conditional;paymentMethod=1)
21. $parameters["CardNumber"] = "4012001045873335";
22. $parameters["ExpiryDateYear"] = "01";
23. $parameters["ExpiryDateMonth"] = "19";
24. $parameters["SecurityCode"] = "123";
25. $parameters["CardHolderName"] = "1";
26. ksort($parameters);
27.
28. //Create an Ordered String of The Parameters Map with Secret Key
29. $orderedString = $SECRET_KEY;
30. foreach($parameters as $param) {
31.     $orderedString .= $param;
32. }
33. echo ("orderdString " . $orderedString);
34.
35. // Generate SecureHash with SHA256
36. $secureHash = hash('sha256', $orderedString, false);
37.

```

7.2 API Approve Request

```
1. $requestQuery = [  
2.     "TransactionID" => $transactionId,  
3.     "MerchantID" => "ANBRedirectM",  
4.     "MessageID" => "1",  
5.     "PaymentMethod" => "1",  
6.  
7.     //for Sadad Payment (conditional paymentMethod => "2")  
8.     // "estn" => "Test123",  
9.  
10.    //for Card Payment (paymentMethod => 1)  
11.    "CardNumber" => "4012001045873335",  
12.    "ExpiryDateYear" => "01",  
13.    "ExpiryDateMonth" => "19",  
14.    "SecurityCode" => "123",  
15.    "CardHolderName" => "1",  
16.    "SecureHash" => $secureHash,  
17.    "PAREs" => urlencode("eJzFV2mTokoW/Ssd9T4a3ewiHZQvkh2"),  
18.  
19. ];  
20. $newRequestQuery = http build query($requestQuery);  
21.  
22. //Send the request  
23. $ch = curl init("https://SR_URL");  
24. curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);  
25. //write parameters  
26. curl_setopt($ch,CURLOPT_POST, true);  
27. curl_setopt($ch,CURLOPT_POSTFIELDS, $newRequestQuery);  
28.  
29. // Get the response  
30. $output = curl_exec($ch);  
31. curl_close($ch);  
32.  
33.  
34. //Output the response  
35. echo ($output);  
36. // this string is formatted as a "Query String" -  
   name=value&name2=value2.....  
37. $result = [];  
38. parse_str($output, $result);  
39. ksort($result);  
40.  
41. // Now that we have the map, order it to generate secure hash and compare it  
   with thereceived one  
42. $responseOrderdString = $SECRET_KEY;  
43. foreach($result as $result_v) {  
44.     $responseOrderdString .= $result_v;  
45. }  
46. echo ("Response Orderd String is " . $responseOrderdString);  
47.  
48. // Generate SecureHash with SHA256  
49. $generatedsecureHash = hash('sha256',$responseOrderdString);  
50.  
51. // get the received secure hash from result map  
52. $receivedSecureHash = $result["Response.SecureHash"];  
53. if($generatedsecureHash !== $receivedSecureHash){  
54.     //IF they are not equal then the response shall not be accepted  
55.     echo ("Received Secure Hash does not Equal generated Secure hash");  
56. }else{  
57.     // Complete the Action get other parameters from result map and do  
   your processes
```

```

58.          // please refer to The Integration Manual to See The List of The
    Received Parameters
59.          $status = $result["Response.Status"];
60.          echo ("Status is :". $status);
61.      }

```

8 Inquiry Message

8.1 Secure Hash Generation

```

1. $SECRET_KEY = "Y2FkMTdlOWZiMzJjMzY4ZGFkMzhkMWIz"; // Use Yours, Please Store Your
    Secret Key in safe Place(e.g. database)
2. // put the parameters in a array to have the parameters to have them sorted
    alphabetically via ksort.
3. $parameters = [];
4. // fill required parameters
5. $parameters['MessageID'] = "2";
6. $parameters['OriginalTransactionID'] = "1440954863817";
7. $parameters["MerchantID"] = "ANBRedirectM";
8. $parameters["Version"] = "1.0";
9.
10.    //Create an Ordered String of The Parameters array with Secret Key
11.    //order parameters alphabatically using ksort
12.    ksort($parameters);
13.    $orderedString = $SECRET_KEY;
14.    foreach($parameters as $param){
15.        $orderedString .= $param;
16.    }
17.    echo ("orderdString ". $orderedString);
18.
19.    // Generate SecureHash with SHA256
20.    $secureHash = hash('sha256', $orderedString, false);
21.

```

8.2 Inquiry Request

```

1.    $requestQueryArr = [
2.        "OriginalTransactionID" => 1440954863817,
3.        "MerchantID" => "ANBRedirectM",
4.        "MessageID" => "2",
5.        "Version" => "1.0",
6.        "SecureHash" => $secureHash,
7.    ];
8.    //generate query string
9.    $newRequestQuery = http_build_query($requestQueryArr);
10.
11.    //Send the request
12.    $ch = curl_init("https://SR_URL");
13.    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
14.
15.    //write parameters
16.    curl_setopt($ch, CURLOPT_POST, true);
17.    curl_setopt($ch, CURLOPT_POSTFIELDS, $newRequestQuery);

```

```

18.
19.     // Get the response
20.     $output = curl_exec($ch);
21.
22.     // To read the output string you might want to split it
23.     // on '&' to get pairs then on '=' to get name and value
24.     // and for a better and ease on verifying secure hash using parse_str
25.     $result = parse_url($output);
26.     //sort result alphanatically
27.     ksort($result);
28.
29.     // Now that we have the map, order it to generate secure hash and compare it
    with the received one
30.     $responseOrderdString = $SECRET_KEY;
31.     foreach($result as $result_v){
32.         $responseOrderdString .= $result_v;
33.     }
34.
35.     echo "Response Orderd String is " . $responseOrderdString;
36.
37.     $generatedsecureHash = hash('sha256',$responseOrderdString);
38.
39.     $receivedSecurehash = $result['Response.SecureHash'];
40.
41.     if($receivedSecurehash != $generatedsecureHash){
42.         echo "Received Secure Hash does not Equal generated Secure hash";
43.     }else{
44.         $status = $result["Response.Status"];
45.         echo "Status is :". $status;
46.     }

```

9 Refund Message

This message is intended to perform a transaction refund. It's based on the Back-To-Back communication model as described in [Communication Model](#) section.

9.1 Secure Hash Generation

```
1.     $SECRET_KEY = "Y2FkMTdlOWZiMzJjMzY4ZGFkMzhkMWIz"; // Use Yours, Please Store
Your Secret Key in safe Place(e.g. database)
2.     // put the parameters in a array to have the parameters to have them sorted
alphabetically via ksort.
3.     $parameters = [];
4.     $transactionId = (int) (microtime(true)*1000); //time in milleseconds
5.
6.     // fill required parameters
7.     $parameters["MessageID"] = "4";
8.     $parameters["TransactionID"] = $transactionId;
9.     $parameters["OriginalTransactionID"] = "1440954863817";
10.    $parameters["MerchantID"] = "ANBRedirectM";
11.    $parameters["Amount"] = "2000";
12.    $parameters["CurrencyISOCODE"] = "840";
13.    $parameters["Version"] = "1.0";
14.
15.    //sort parameters alphabatically
16.    ksort($parameters);
17.    //Create an Ordered String of The Parameters Map with Secret Key
18.
19.    $orderedString = $SECRET_KEY;
20.    foreach ($parameters as $param) {
21.        $orderedString .= $param;
22.    }
23.    echo "orderdString ". $orderedString;
24.
25.    // Generate SecureHash with SHA256
26.    $secureHash = hash('sha256', $orderedString, false);
27.
```

9.2 Refund Request

```
1.     $requestQueryArr = [
2.         "TransactionID" => $transactionId,
3.         "MarchantID" => "ANBRedirectM",
4.         "MessageID" => "4",
5.         "Amount" => "2000",
6.         "OriginalTransactionID" => "1440954863817",
7.         "CurrencyISOCODE" => "840",
8.         "SecureHash" => $secureHash,
9.         "Version" => "1.0"
10.    ];
11.    //generate query string
12.    $newRequestQuery = http_build_query($requestQueryArr);
13.
14.    //Send the request
15.    $ch = curl_init("https://SR_URL");
16.    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
17.
18.    //write parameters
19.    curl_setopt($ch, CURLOPT_POST, true);
20.    curl_setopt($ch, CURLOPT_POSTFIELDS, $newRequestQuery);
```

```

21.
22.     // Get the response
23.     $output = curl_exec($ch);
24.     curl_close($ch);
25.
26.
27.     //Output the response
28.     echo ($output);
29.     // this string is formatted as a "Query String" -
name=value&name2=value2.....
30.     $result = [];
31.
32.     // To read the output string you might want to split it
33.     // on '&' to get pairs then on '=' to get name and value
34.     // and for a better and ease on verifying secure hash using parse_str
35.     parse_str($output, $result);
36.
37.     //sort result alphanatically by key
38.     ksort($result);
39.
40.     // Now that we have the array sorted, order it to generate secure hash and
compare it with the received one
41.
42.     $responseOrderdString = $SECRET_KEY;
43.     foreach ($result as $result_v) {
44.         $responseOrderdString .= $result_v;
45.     }
46.     echo "Response Orderd String is " . $responseOrderdString;
47.     // Generate SecureHash with SHA256
48.     $generatedsecureHash = hash('sha256',$responseOrderdString);
49.     // get the received secure hash from result map
50.     $receivedSecurehash = $result["Response.SecureHash"];
51.     if($receivedSecurehash != $generatedsecureHash){
52.         //IF they are not equal then the response shall not be accepted
53.         echo ("Received Secure Hash does not Equal generated Secure hash");
54.     }else{
55.         // Complete the Action get other parameters from result map and do
your processes
56.         // please refer to The Integration Manual to See The List of The
Received Parameters
57.         $status = $result["Response.Status"];
58.         echo ("Status is :". $status);
59.     }

```