

**PaymentGatewayDataPreprocess**













**paymentgatewaycallback**

Step 1:

The end point

<https://tc9gXXXXnjf.execute-api.us-east-1.amazonaws.com/dev/paymentgatewaydatapreprocessor>

Request Type:

POST

Payload:

{

“amount”:””,

“email”:””,

“first\_name”:””,

“phone”:””

}

Step 2:

A lambda **PaymentGatewayDataPreprocess** will be invoked and will set furl,key,productinfo.service\_provider,surl,txnid

the following payload will be created.

Payload:

{

“amount”:””,

“email”:””,

“first\_name”:””,

“phone”:””,

“furl,key”:””,

“productinfo”:””,

“service\_provider”:””,

“surl”:””,

“txnid”:””

}

Step 3:

This payload/information will be saved in database

Step 4,5,6:

The payload will be delivered to client/user.

Step 7:

a virtual page will be generated will all the required fields and submitted to PayU server then

User gets redirect to PayU money payment gateway.

User will now initiate with the payment and will pay the required amount.

Step 8:

Here transaction status based on success/failure will be send by PayU to the below endpoint.

<https://XXXXX.execute-api.us-east-1.amazonaws.com/dev/>

Step 9:

calculate hash with the returning item from the PayU.

Match the hash given by PayU to check weather data is authentic or is not tampered.

reference:

<https://www.payumoney.com/dev-guide/webcheckout/redirect.html>

Hash for Normal Payment Response:

salt|status||||||udf5|udf4|udf3|udf2|udf1|email|firstname|productinfo|amount|txnid|key

below is the success sample payload of PayU money and now with the below data I must check its authenticity.

Ex.

{    
   **"split\_info"**:"XXX17743",  
   **"customerName"**:"Test user",  
   **"additionalCharges"**:"",  
   **"paymentMode"**:"DC",  
   **"hash"**:"XXX00c3a13f37c1271df8c0ebe67c4aad2d29a2086e80aaa3d16580bbe38f9ffd0c3996eab241aa730a4efe512c6c15730ca66020064d71d85dcb68631119f29",  
   **"status"**:"Success",  
   **"error\_Message"**:"No Error",  
   **"paymentId"**:"59017743",  
   **"productInfo"**:"Description1",  
   **"customerEmail"**:"storedcard8@yopmail.com",  
   **"customerPhone"**:"6709133497",  
   **"merchantTransactionId"**:"XXX6753-59017743",  
   **"amount"**:"100.0",  
   **"udf2"**:"",  
   **"notificationId"**:"37208",  
   **"udf1"**:"",  
   **"udf5"**:"",  
   **"udf4"**:"",  
   **"udf3"**:""  
}

Here the salt ie private key is missing. (it’s missing and confidential, since its not given in the payload and to create hash that exact salt key required. Hence no one can tamper that data without salt key )

Here now check the authenticity of data:

Create hash string first with the given data in following sequence provide your salt key given by PayU and take rest of the things from response:

salt|status||||||udf5|udf4|udf3|udf2|udf1|email|firstname|productinfo|amount|txnid|key

fE0aTrjr|success||||||||||| storedcard8@yopmail.com |Test user| Description1|100.0|4826753-59017743|UFu3ed

now generate hash for the string

fE0aTrjr|success||||||||||| storedcard8@yopmail.com |Test user| Description1|100.0|4826753-59017743|UFu3ed

now say the hash code u get is

64700c3a13f37c1271df8c0ebe67c4aad2d29a2086e80aaa3d16580bbe38f9ffd0c3996eab241aa730a4efe512c6c15730ca66020064d71d85dcb68631119f29

Now check this hash code with the given code by PayU if both are same the data is vailid and not tampered.

Step 10:

Save data in database.

Send 11,12,13:

Reply user with appropriate message

Payment success/failure.