Paytm Integration Guide for App (Android)

Contents

Merchant App

Paytm Android SDK should be bundled with merchant's App to complete Paytm's PG [Credit/Debit Card, Net-banking] and Wallet integration. Merchant App will show user option to select Paytm as one of the payment option and fetch necessary details to SDK for completing the payment.

In SDK

The SDK will handle all the user flows as below:

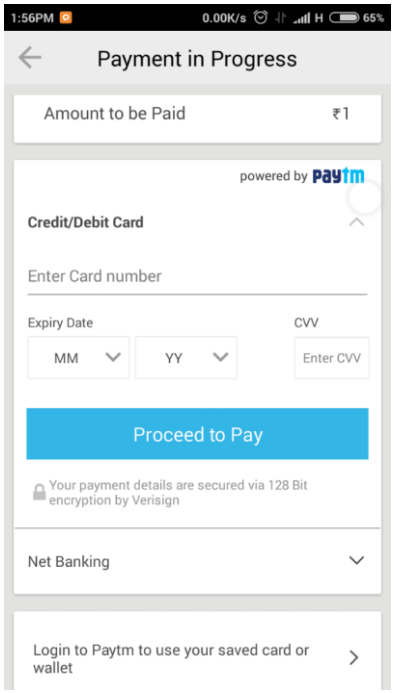
- Payment Option selection
- Showing the Amount including convenience charges based on Payment option selected
- Card based:
    - Validating card number and generating secure token
    - Redirecting user to bank's 3D secure page for completing the payment

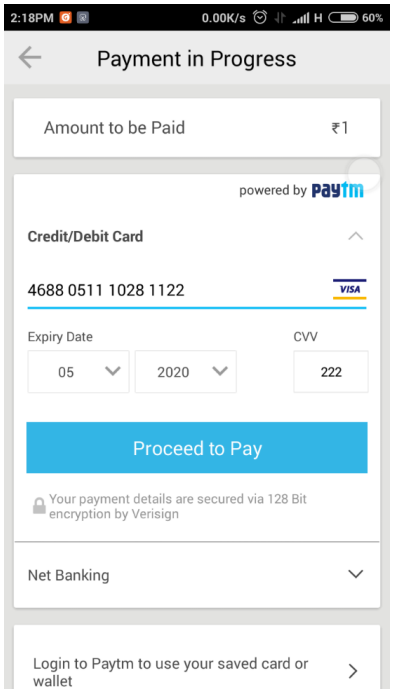
- Net-banking based:
    - Displaying list of banks available for Net--banking
    - Redirecting user to bank's page

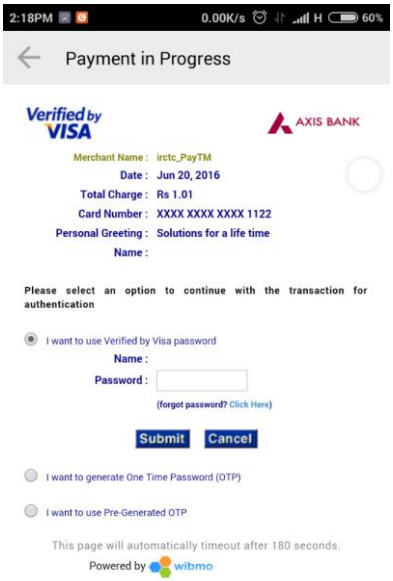
- Wallet based:
    - Assist user to login in Paytm Wallet by auto-reading the OTP
    - Check user's available balance
    - Fetching user's saved cards at Paytm
    - Allowing user to add differential amount in Paytm Wallet by card or net-banking
    - Redirecting user to bank's 3D secure/ Net-banking page in case of insufficient balance in Paytm Wallet

- Completing the payment from one of the above options
- Redirecting back to app from bank's page
- Closing the web-view and posting the result back to app and merchant server

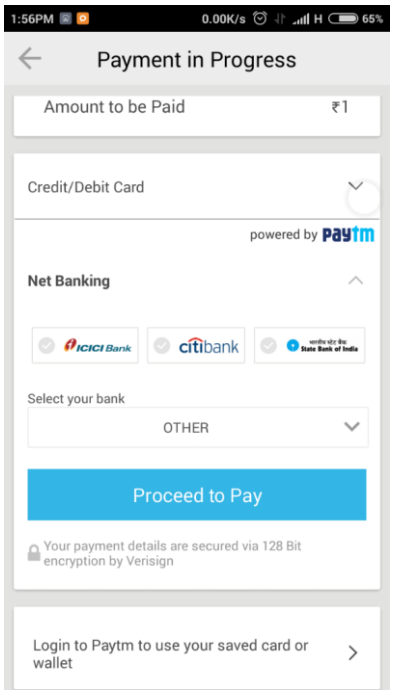
Payment via Card:

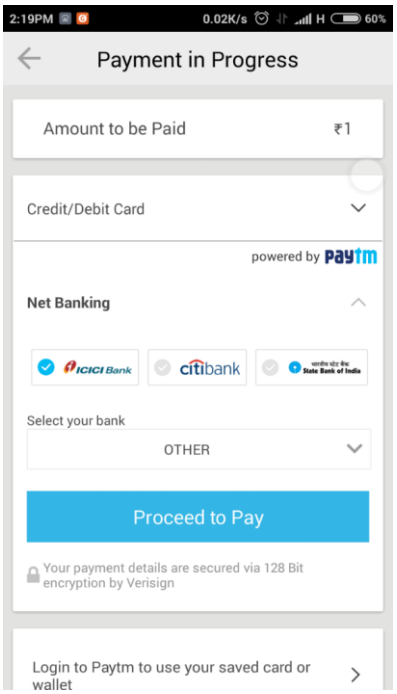


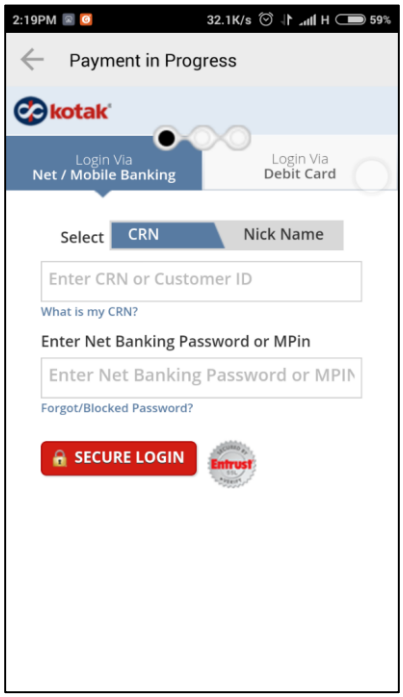




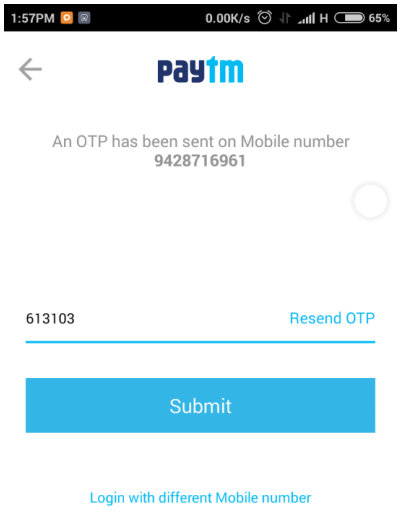
Payment via Net-banking

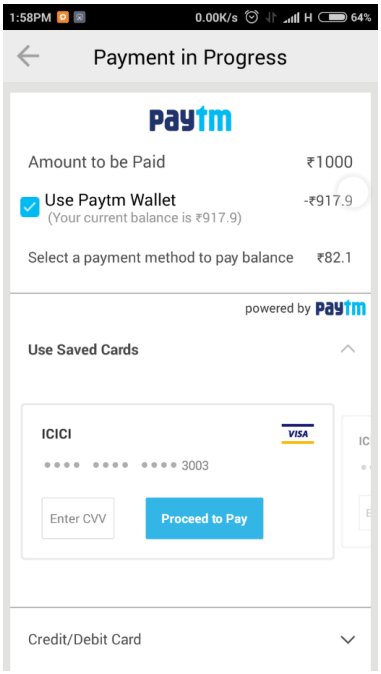






Payment via Wallet - Login with Wallet





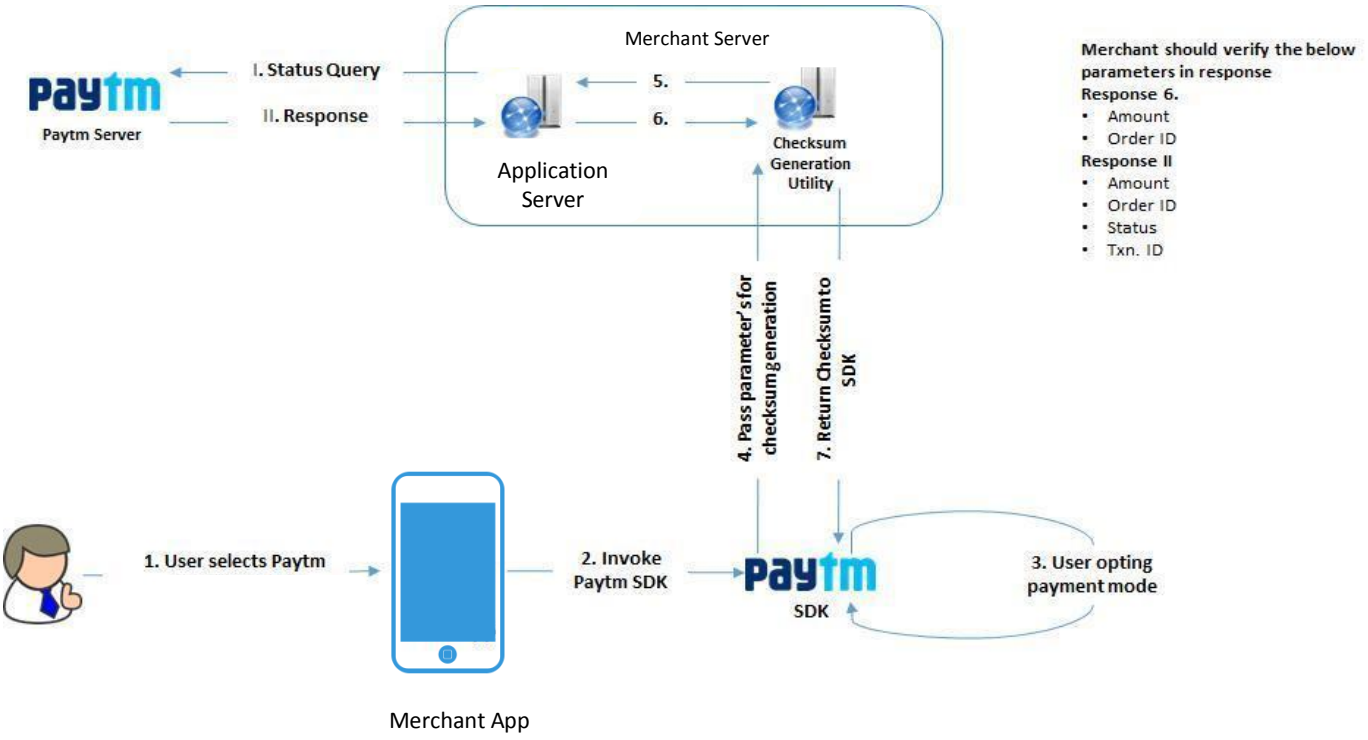
Merchant Server

The merchant needs to deploy checksum logic at their server to generate and verify Checksum. Checksum helps ensure the integrity of transactions i.e. requests have not been intercepted and tampered with. Paytm provides Checksum utilities for different development platforms.
SDK would call the Merchant server for generating the checksum at multiple stages of payment. On this server component, merchant must validate the amount against the order ID provided by SDK from the actual amount and other parameters as well for all these calls.
SDK would call this component and fetch checksum for following:

- For submitting payment request to Paytm for card or net---banking payment,
- For generating OTP for logging in Paytm Wallet,
- For payment request to Paytm for Wallet– add and pay or
- For payment request to Paytm for Wallet– with sufficient balance

Merchant server would need to validate the result with Paytm through server---to---server call and accordingly provide the result back to customer and process the request. Merchant server would

Flow diagram:



Prerequisites for using SDK.

Staging And Production Credentials: Paytm will provide following credentials:

1. **A Unique Merchant ID** – This ID is unique for each merchant.
2. **Website**: A website name.
3. **Industry Type ID**: An ID for the Industry Type.
4. **Paytm Merchant Key**: This is a unique key, which will be needed for generating & verifying checksums. This is a secret key and should never be passed along with request parameters in API calls to Paytm Servers.
5. **Channel id**: An Id for the channel being used.

Checksum generation server URL and callback URL:

The merchant needs to set up a server at their end to generate a checksum and host a callback URL for checksum validation.

1. In generate checksum API call; merchants must verify the transaction amount ("TXN_AMOUNT" in request) with their order amount (total order amount for the items selected by a customer). In case of any mismatch, return null instead of checksum.
2. At the end of the transaction or before fulfilling the order; merchant must call the Transaction Status API to validate the transaction final status and amount.

Configurations: Add Permissions

Add the following permissions to AndroidManifest.xml.

```
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
```

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<protected-broadcast android:name="android.intent.action.MEDIA_MOUNTED"/>

Optional( To make otp autofill)
<uses-permission android:name="android.permission.READ_SMS"/>
```

NOTE: If you face a build error while importing aar as module please do this

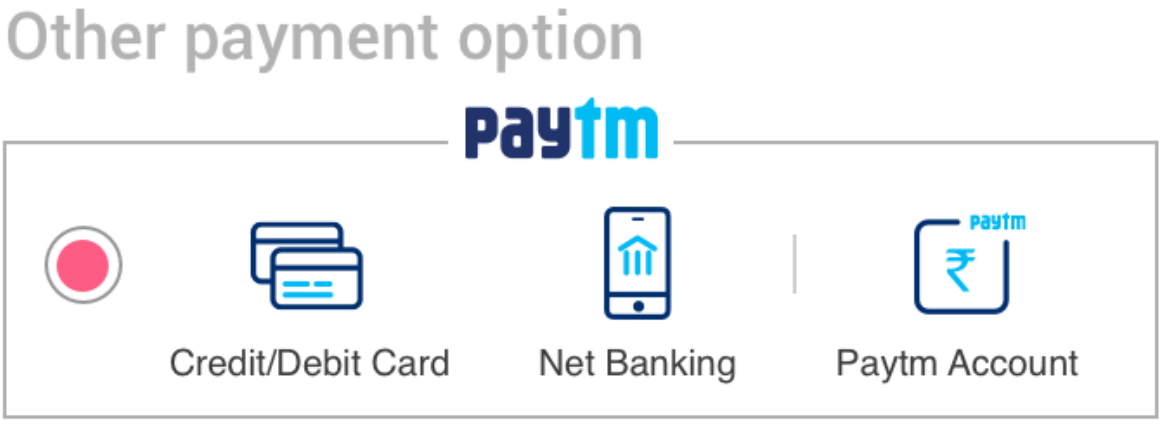
```
tools:replace:"android:theme"
```

Step 1: Generate a bill

Before initiating the payment flow, make sure you have a bill generated by your server-side code. This bill should have at least unique Order ID.

Step 2: Add a Payment Option

Inside your payment checkout flow, add another payment option and once a user clicks on it, Paytm SDK should be invoked.



Step 3: Paytm SDK flow

App should instantiate a map of mandatory parameters for processing the request. The keys used in the map are constants as shown in the code snippet below. The values of the key---value pair in the map are sample values. These values are explained after the code snippet.

```
LinkedHashMap<String, String> paramMap = new
LinkedHashMap<String, String>();
paramMap.put("ORDER_ID", "order12345")
paramMap.put("MID","klbGlV59135347348753";
paramMap.put("CUST_ID",'CUST110')
paramMap.put("CHANNEL_ID", "WAP");
paramMap.put("INDUSTRY_TYPE_ID","Retail");
paramMap.put("WEBSITE", "paytm");
paramMap.put("TXN_AMOUNT", "1.0");
paramMap.put("THEME ", "merchant");
paramMap.put("MSISDN ", "mobile number");
paramMap.put("EMAIL ", "email ID");
paramMap.put("MERC_UNQ_REF ", "meta data");
```

where, the *values* in the above map are as:

- ORDER_ID is the order id of that particular order and must be unique for every hit.
- MID is the merchant id of the merchant and its mandatory .
- CUST_ID is the customer id of merchant.
- CHANNEL_ID is provided by Paytm
- INDUSTRY_TYPE_ID is the industry type id provided by Paytm.
- WEBSITE is the constant provided by Paytm .
- TXN_AMOUNT is the total amount, which will be paid via Paytm .
- THEME is provided by Paytm
- MSISDN is customer mobile number.
- EMAIL is customer email ID.
- MERC_UNQ_REF is the meta data field

Instantiate a *PaytmMerchant* object which will contain the Check Sum generation and validation URLs

```
PaytmMerchant merchant = new PaytmMerchant(String

inChecksumGenerationURL, String inChecksumVerificationURL);
```

where,
inChecksumGenerationURL is the Merchant Server URL which will be called by SDK for generating checksum at multiple steps and
inChecksumVerificationURL is the Merchant Server URL which will be called by SDK for verifying the checksum in response after payment completion

Once the above objects are initiated, the Paytm native flow will be invoked by the following method call:

```
PaymentSdkMainClass paymentSdkMainClass = new
PaymentSdkMainClass(this);
/*
 @enableCC  true when you want credit/debit card payment else false
 @enableNB  true when you want NetBanking else false
 @enableWallet true when you want paytmwallet as option else false
 @colorCode string representing hexadecimal color code.
*/
paymentSdkMainClass.setDyanamicValues(enableCC,enableNB,enableWallet
,colorCode);
paymentSdkMainClass.pay(paramMap, merchant,mHideArrow null,
isStaging , allowPaytmAssist, new PaytmPaymentTransactionCallback()
{

        @Override

        public void someUIErrorOccurred(String
          arg0) { // TODO client can handle
          response

        }

        @Override
```

```
        public void onTransactionSuccess(Bundle
          arg0) { // TODO transaction success

        }
        @Override

        public void onTransactionFailure(String arg0, Bundle
          arg1) { // TODO transaction failure

        }

        @Override

        public void onErrorLoadingWebPage(int arg0, String arg1, String
          arg2) { // TODO show any error message

        }

        @Override

        public void onBackPressedCancelTransaction()
          { // TODO client can show a confirmation
          dialog.

        }

        @Override

        public void networkNotAvailable() {

          // TODO client can show network not available dialog or ask for retry

        }

        @Override

        public void clientAuthenticationFailed(String arg0) {

          // TODO  this shows either checksum is not valid or authentication failed

        }

});
```

In this call the parameters used are following:

- ***paramMap*** is the linkedhashmap containing all the value params needed for transaction.
- ***merchant*** is the merchant object which we initialize with checksum verify url and checksum validate url.
- third param is for certificate which you can send null always.
- ***isStaging*** is a boolean for pointing app to staging(true) or development(false).

Step 4: Get back to your app

Paytm SDK webview call back:

In case of webview flow, the callback methods are already passed in the `paymentSdkMainClass.pay()` method. Following are the methods which you can override with your custom implementation .

- *onTransactionSuccess(String arg0)*. can implement a success dialog message or success page.
- *someUIErrorOccured(String arg0)* can handle for any ui related error.
- *onTransactionFailure(String arg0)* can handle for error if transaction is not successfull.
- *onErrorLoadingWebPage(int arg0, String arg1, String arg2)* if you gets any error while loading webpage then show any error.
- *onBackPressedCancelTransaction()* if user presses backpress then you can show a confirmation dialog.
- *networkNotAvailable()* if network is not available you can show for retry or connect to a network.
- *clientAuthenticationFailed(String arg0)* handle for checksum is not valid or authentication failed.

Step 5: Status Query for Transactions

To enable merchants to get or verify the transaction status, we are providing the transaction status tracking URL. It is basically a servlet which accepts merchant id (MID) and merchant's order id (ORDERID) and retrieves the transaction status from the database. These parameters are to be posted through application over URL connection using SSL.

Status Query API for production environment:
https://secure.paytm.in/oltp/HANDLER_INTERNAL/TXNSTATUS
Status Query API for staging environment:
https://pguat.paytm.com/oltp/HANDLER_INTERNAL/TXNSTATUS

Request Parameters: (Should be passed as JSON string in "JsonData" parameter name)

S.No	Parameter Name	Description	Type	Length	Mandatory
1.	MID	This is a unique merchant Id provided to merchant by Paytm at the time of merchant creation.	Alphanumeric		Yes
2.	ORDERID	This is the application transaction Id that was sent by merchant to Paytm at the time of transaction request.	Alphanumeric	50	Yes

Response Parameters: (Response will come as JSON string)

S.no	Parameter Name	Description	Type	Mandatory
1.	TXNID	This is a unique Paytm transaction Id that is issued by Paytm for each valid transaction request received from the merchant.	Numeric	Yes
2.	BANKTXNID	The transaction Id sent by the bank (NULL or empty string if the transaction doesn't reaches to the bank).	Alphanumeric	Yes
3.	ORDERID	This is the application transaction Id that was sent by merchant to Paytm at the time of transaction request.	Alphanumeric	Yes
4.	TXNAMOUNT	Amount of transaction.	Numeric	Yes
5.	STATUS	This contains the transaction status and has only two values: TXN_SUCCESS TXN_FAILURE	Alphanumeric	Yes
6.	TXNTYPE	Any one of below values: SALE	Alphanumeric	Yes
7.	GATEWAYNAME	The gateway used by Paytm (ICICI/CITI/WALLET etc).	Alphanumeric	Yes
8.	RESPCODE	This is a numeric transaction response code. All codes refer to a transaction failure or success with each code representing a different reason for failure. Refer to Annexure A for full list.	Alphanumeric	Yes

9.	RESPMSG	This contains a short description of the transaction status. In case of a failed transaction the message will describe the potential reason for the failure.	Alphanumeric	Yes
10.	BANKNAME	Bank name of the card issuing bank.	Alphanumeric	Yes
11.	MID	This is a unique merchant Id provided to merchant by Paytm at the time of merchant creation.	Alphanumeric	Yes
12.	PAYMENTMODE	Mode of Payment. ● CC ● DC ● NB ● IMPS ● PPI	Alphanumeric	Yes
13.	REFUNDAMT	Total amount refunded till now if merchant has raised any requests.	Numeric	Yes
14.	TXNDATE	Date of transaction.	DateTime	Yes

Response JSON String:

{"TXNID":"62284943","BANKTXNID":"099172","ORDERID":"Test87984","TXNAMOUNT":"1","STATUS":"TXN_SUCCESS","TXNTYPE":"SALE","GATEWAYNAME":"CITI","RESPCODE":"01","RESPMSG":"Txn Successfull.","BANKNAME":"HDFC","MID":"xxxxx34213145601111","PAYMENTMODE":"CC","REFUNDAMT":"1","TXNDATE":"2012-11-09 02:10:29.742447"}

How to generate or validate checksum?

Checksum Generation

The checksum ensures integrity of the transaction, and is generated using the secret Merchant Key that is issued by Paytm. Merchant needs to host the checksum generation file at their own server.
Follow the below given steps to host the file:
Step 1 *Download the jars from this location. Click here.*
Step 2 *Configure jars in your project.*
Step 3 *Host the following code in your server (for JSP).*

```
<%@ page import="com.google.gson.GsonBuilder"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>
<%@page import="com.google.gson.Gson"%>
<%@ page import="java.util.*" %>
<%@ page import="java.net.URLEncoder" %>
<%@ page import="com.paytm.pg.merchant.CheckSumServiceHelper"%>
<%


Enumeration<String> paramNames = request.getParameterNames();
Map<String, String[]> mapData = request.getParameterMap();
TreeMap<String,String> parameters = new TreeMap<String,String>();
String checkSum = "";
boolean isAppExist = false;

if(mapData.containsKey("isAppExist")){
  if(mapData.get("isAppExist")[0].equals("true")){
    isAppExist = true;
  }
}

if(isAppExist){
        checkSum =  CheckSumServiceHelper.getCheckSumServiceHelper().genrateCheckSum("Paytm
Merchant Key", mapData.get("jsonData")[0]);
}else{
        while(paramNames.hasMoreElements()) {
          String paramName = (String)paramNames.nextElement();
          if(paramName.equals("isAppExist"))
            continue;

            parameters.put(paramName,mapData.get(paramName)[0]);
        }
        checkSum =  CheckSumServiceHelper.getCheckSumServiceHelper().genrateCheckSum("Paytm
Merchant Key", parameters);
}
parameters.put("CHECKSUMHASH",checkSum);
parameters.put("payt_STATUS","1");
Gson gson = new GsonBuilder().disableHtmlEscaping().create();
out.println(gson.toJson(parameters));
%>
```

Checksum Validation

To get access for server side Utilities for checksum generation and validation, click here.

How to refund the transaction?

If the merchant, for any reason, wants to cancel or refund a success transaction, then it can initiate refund either by our Merchant Panel or by using our Refund API.

Merchant Panel

Paytm's merchant panel would be provided by the on-boarding team refund and reporting purposes.

Refund API

Basic information needs to be sent across with the refund request. Following are the refund APIs and their request and response parameters.

Refund API for production environment:
https://secure.paytm.in/oltp/HANDLER_INTERNAL/REFUND
Refund API for staging environment:
https://pguat.paytm.com/oltp/HANDLER_INTERNAL/REFUND
Request Parameters: (Should be passed as JSON string in "JsonData" parameter name)

S.no	Parameter Name	Description	Type	Mandatory
1	MID	This is a unique merchant Id provided to merchant by Paytm at the time of merchant creation.	Alphanumeric	Yes
2	TXNID	This is an unique Paytm transaction Id that is issued by Paytm for each valid transaction request received from the merchant.	Numeric	Yes
3	ORDERID	This is the application transaction Id that was sent by merchant to Paytm at the time of transaction request.	Alphanumeric	Yes
4	REFUNDAMOUNT	Amount to be refunded.	Numeric	Yes

5	TXNTYPE	Any one of below values: REFUND CANCEL	Alphanumeric	Yes
6	COMMENTS	Any comments can be given here.	Alphanumeric	No
7	REFID	Unique ID for every refund request sent by merchant to Paytm	Alphanumeric	Yes
8	CHECKSUM	Checksum calculated based on pre-defined logic. Checksum is used to ensure data is not tempered when request is posted on the Paytm URL. In case of checksum mismatch due to data tempering Paytm will reject the transaction.	Alphanumeric	Yes

Response Parameters: (Response will come as JSON string)

S.no	Parameter Name	Description	Type	Mandatory
1	MID	This is a unique merchant Id provided to merchant by Paytm at the time of merchant creation.	Alphanumeric	Yes
2	TXNID	This is a unique Paytm transaction Id that is issued by Paytm for each valid transaction request received from the merchant.	Numeric	Yes
3	ORDERID	This is the application transaction Id that was sent by merchant to Paytm at the time of transaction request.	Alphanumeric	Yes
4	TXNAMOUNT	Amount of transaction.	Numeric	Yes
5	REFUNDAMOUNT	Amount to be refunded.	Numeric	Yes
6	TXNDATE	Date of transaction.	DateTime	Yes
7	RESPCODE	This is a numeric transaction response code. All codes refer to a transaction failure or success with each code representing a different reason for failure. Refer to Annexure A for full list.	Alphanumeric	Yes

8	RESPMSG	This contains a short description of the transaction status. In case of a failed transaction the message will describe the potential reason for the failure.	Alphanumeric	Yes
9	STATUS	This contains the transaction status and has only two values: • TXN_SUCCESS • TXN_FAILURE	Alphanumeric	Yes
10	REFID	Unique ID for every refund request sent by merchant to Paytm	Alphanumeric	Yes
11	REFUNDID	Unique ID for every successful refund sent by Paytm in response to refund request	Alphanumeric	No

Status Query for Refunds

Status Query API for Refund on production environment:
https://secure.paytm.in/oltp/HANDLER_INTERNAL/REFUND_STATUS
Status Query API for Refund on staging environment:
https://pguat.paytm.com/oltp/HANDLER_INTERNAL/REFUND_STATUS

Request Parameters: (Should be passed as JSON string in "JsonData" parameter name)

S.no	Parameter Name	Description	Type	Length	Mandatory
1	MID	This is a unique merchant Id provided to merchant by Paytm at the time of merchant creation.	Alphanumeric		Yes
2	ORDERID	This is the application transaction Id that was sent by merchant to Paytm at the time of transaction request.	Alphanumeric	50	Yes
3	REFID	This is reference ID shared while raising Refund.	Alphanumeric	50	Yes

Response Parameters: (Response will come as JSON string)

S.no	Parameter Name	Description	Type
1	TXNID	This is a unique Paytm transaction Id that is issued by Paytm for each valid transaction request received from the merchant.	Numeric
2	BANKTXNID	The transaction Id sent by the bank (NULL or empty string if the transaction doesn't reaches to the bank).	Alphanumeric
3	ORDERID	This is the application transaction Id that was sent by merchant to Paytm at the time of transaction request.	Alphanumeric
4	TXNAMOUNT	Amount of transaction.	Numeric
5	STATUS	This contains the transaction status and has only two values: • **TXN_SUCCESS** • **TXN_FAILURE**	Alphanumeric
7	GATEWAY	The gateway used by Paytm (ICICI/CITI/WALLET etc).	Alphanumeric
8	RESPCODE	This is a numeric transaction response code. All codes refer to a transaction failure or success with each code representing a different reason for failure. Refer to Annexure A for full list.	Alphanumeric
9	RESPMSG	This contains a short description of the transaction status. In case of a failed transaction the message will describe the potential reason for the failure.	Alphanumeric
10	MID	This is a unique merchant Id provided to merchant by Paytm at the time of merchant creation.	Alphanumeric
11	PAYMENTMODE	Mode of Payment. 1. CC 2. DC 3. NB 4. IMPS 5. PPI	Alphanumeric
12	REFUNDAMOUNT	Refund amount as received in the request	Numeric
13	TOTALREFUNDAMT	Total amount refunded till now if merchant has raised any requests.	Numeric
14	TXNDATE	Date of transaction.	DateTime

15	REFUNDDATE	Date of refund	DateTime
16	REFUNDTYPE	Type of Refund Issued	Alphanumeric
17	REFID	This is reference ID shared while raising Refund.	Alphanumeric
18	REFUNDID	Unique refund id generated at Paytm end	Alphanumeric