# API HOÀN HUỶ KHI THANH TOÁN

1. **Dữ liệu đối tác cần chuẩn bị.**

String cmd // Mã lệnh. Giá trị cố định là: REFUND\_PAYMENT

String merchant\_code

String access\_code // Mã truy cập do Viettel cấp cho đối tác

String hash\_key // Khóa bảo mật để mã hoá dữ liệu do Viettel cấp cho đối tác

String order\_id

String originalRequestId // Mã giao dịch thanh toán tương ứng bên Viettel

String refundType // Hình thức hoàn tiền: 0 hoàn toàn phần, 1 hoàn một phần

// Hiện nay chỉ hỗ trợ hoàn toàn phần

Double trans\_amount // Số tiền hoàn = số tiền thanh toán

String reason\_content // Lý do hoàn tiền

String version // Giá trị cố định 2.0

String check\_sum // Chuỗi mã hoá SHA1 -> encode base64 -> utf8 từ dữ liệu của:

// access\_code + cmd + merchant\_code + order\_id + originalRequestId + refundType + trans\_amount + version

//Tạo check sum

String date = access\_code + cmd + merchant\_code + order\_id + originalRequestId + refundType + trans\_amount + version;

check\_sum=Security.hmac(data, hash\_key);

1. **Chuẩn bị các param để thực hiện request tới viettel**

// Cần khai báo thêm 2 dependencies là HttpClient và Fluent HC vào file pom nếu chưa có. Tham khảo thêm tại http://hc.apache.org/httpcomponents-client-ga/quickstart.html

CloseableHttpClient httpclient = HttpClients.createDefault();

HttpPost httpPost = new HttpPost("http://sandbox.viettel.vn/PaymentAPI/webresources/postData"); // địa chỉ sẽ do viettel cung cấp để thực hiện request tới, đây là địa chỉ của máy chủ test của viettel

httpPost.addHeader("Content-Type", "application/x-www-form-urlencoded");

List<NameValuePair> nvps = new ArrayList<NameValuePair>(); //Khai báo NameValuePair(nvps) để truyền đủ các param cần thiết(cmd,merchant\_code... Như trong tài liệu kết nối thanh toán đã mô tả) để thực hiện Post request qua viettel

nvps.add(new BasicNameValuePair("cmd", cmd));

nvps.add(new BasicNameValuePair("merchant\_code", merchant\_code));

nvps.add(new BasicNameValuePair("order\_id", order\_id));

nvps.add(new BasicNameValuePair("originalRequestId", originalRequestId));

nvps.add(new BasicNameValuePair("refundType", refundType));

nvps.add(new BasicNameValuePair("trans\_amount", trans\_amount));trans\_content

nvps.add(new BasicNameValuePair("trans\_content", trans\_content));

nvps.add(new BasicNameValuePair("version", version));

nvps.add(new BasicNameValuePair("check\_sum", check\_sum));

httpPost.setEntity(new UrlEncodedFormEntity(nvps));

CloseableHttpResponse response2 = httpclient.execute(httpPost);//Thực hiện(execute) post request và nhận response chuẩn bị lấy dữ liệu viettel trả về

1. **Nhận dữ liệu từ JSON trả về của viettel**

try{

System.out.println(response2.getStatusLine());

HttpEntity entity2 = response2.getEntity(); //Chuyển response thành entity

// Lấy dữ liệu từ response body

String retSrc = EntityUtils.toString(entity2); //Chuyển entity sang String

// parsing JSON

JSONObject result = new JSONObject(retSrc); // Chuyển String sang JSON Object

//Dữ liệu viettel trả về từ json

String merchant\_code\_res = result.get("merchant\_code");

String order\_id\_res = result.get("order\_id");

String error\_code\_res = result.get("error\_code"); // Thành công sẽ trả về dạng 00. Trường hợp khác kiểm tra trong tài liệu

String error\_msg\_res = result.get("error\_msg"); // Mô tả khái quát nội dung lỗi

String merchant\_request\_id\_res = result.get("merchant\_request\_id"); // Mã giao dịch huỷ bên Viettel

String refund\_request\_id\_res = result.get("refund\_request\_id"); // Mã giao dịch hoàn tiền cho KH

String version\_res = result.get("version"); // Phiên bản kết nối. Giá trị cố định là: 2.0

String check\_sum\_res = result.get("check\_sum"); // Chuỗi mã hoá SHA1 -> encode base64 -> utf8 từ dữ liệu của:

// access\_code + error\_code + merchant\_code + order\_id + version

//Tạo check sum bên hệ thống đối tác có thể dùng vào so sánh với checksum viettel để xác thực dữ liệu 2 bên

String data\_doitac = access\_code + error\_code + merchant\_code + order\_id + version;

String check\_sum\_doitac = Security.hmac(data\_doitac,hash\_key);

EntityUtils.consume(entity2);

} catch (Exception e) {

System.out.println(e);

} finally {

response2.close();

}