بسم تعالی

راهنمای راه اندازی

درگاه پرداخت اینترنتی

نسخه ۱/۷ - اردیبهشت ۱۳۹۷

نسخه ۷,۱ – اردیبهشت ماه ۱۳۹۶

نویسنده: فرشید گیلک

www.asanpardakht.ir

تغییرات نسخ این سند:

شماره نسخه	تاریخ انتشار	تغییرات نسخه
۲	۱۳۹۱/۱۲/۷	نسخه اولیه سند درگاه پرداخت اینترنتی جدید (شاپرکی) آسان پرداخت پرشین.
۲,۱	۱۳۹۱/۱۲/۱۴	راهنمای انتقال از درگاه پرداخت به سایت پذیرنده افزوده شد.
۲,۲	۱۳۹۱/۱۲/۲۰	آدرس های وب سرویس ها بهبود داده شد تا پذیرندگان نیاز به فراخوانی وب سرویس از دو WSDL مجزا نداشته باشند.
۲,۳	۱۳۹۲/۱/۱۸	دو فیلد تکمیلی جدید از سوی درگاه به سایت پذیرنده بازگردانده می شوند. اطلاعات این دو فیلد در سند آورده شده است. در پارامترهای بازگردانده شده از سوی درگاه پرداخت به سایت پذیرنده SalesReferenceID به PayGateTranID تغییر نام یافت. شرحی در رابطه با رمزگشایی از رشته پاسخ درگاه پرداخت به سایت پذیرنده آورده شد.
۲,۴	۱۳۹۲/۴/۲۵	شیوه ارجاع مشتری به درگاه پرداخت بصورت شفاف تری توضیح داده شد. اطلاعات بیشتری در رابطه با پارامترهای بازگشتی از درگاه پرداخت به وب سایت پذیرنده ارائه شد. شرحی در رابطه با مراحل گام به گام انجام تراکنش با درگاه پرداخت ارائه شد.
۲,۵	۱۳۹۲/۵/۲۷	آدرس ایمیل مکاتبات درگاه پرداخت تغییر یافت. آدرس ذکر شده در صفحه ۱۴، اصلاح شد.
۳,۰	۱۳۹۲/۸/۲۵	درگاه پرداخت روی زیر دامنه شاپرک به عنوان جایگزین درگاه فعلی ارائه شد.
۴,۱	۱۳۹۳/۶/۷	پشتیبانی از تقسیم وجوه به درگاه پرداخت اینترنتی افزوده شد.
۴,۲	۱۳۹۴/۶/۷	شرح بیشتری در خصوص خطاهای درگاه پرداخت به مستند اضافه شد.
۴,۳	۱۳۹۴/۱۰/۲۰	رفع اشکالات و اضافه شدن توضیحات در خصوص روال های جدید احراز هویت پذیرنده
۴,۵	۱۳۹۵/۴/۱۹	حذف متد تسویه دسته ای، افزوده شدن متد بررسی وضعیت تراکنش، برخی اصلاحات دیگر
۵	۱۳۹۵/۱۱/۱	افزودن روشی جدید برای دریافت نتیجه تراکنش صرفا با داشتن شماره فاکتور محلی
۶	۱۳۹۶/۴/۱۳	افزودن روش پرداخت بدون توکن و سرویس ذخیره کارت های مشتریان روی درگاه پرداخت
۶,۱	۱۳۹۶/۶/۲۶	افزودن مثال های رمزنگاری به زبان جاوا
۶,۲	۱۳۹۶/۹/۰۶	افزودن کنترل های تکمیلی آدرس کال بک پذیرنده برای اعلان نتیجه تراکنش
۷	۱۳۹۷/۱/۲۸	افزودن متد ملغی کردن Verification معرفی سرویس انتخاب کارت در هنگام پرداخت بازنشسته شدن سرویس سابق ذخیره و بازیابی کارت
۷,۱	۱۳۹۷/۲/۰۱	بروز رسانی کدهای خطا در عملیات پس از پرداخت

فهرست مطالب

الزامات شاپرک برای متقاضیان درگاه های پرداخت اینترنتی

الزام شاپرک برای متقاضیان درگاه های پرداخت اینترنتی عبارت است از اخذ نماد اعتماد الکترونیکی: اکنون اخذ نماد الکترونیکی از حالت اختیاری خارج شده و بصورت پذیرندگان باید اقدام به اخذ این نماد نمایند. اطلاعات بیشتر در این رابطه را می توانید در این وب سایت مشاهده نمایید:

http://www.enamad.ir

مقدمه

برای راحتی کاربران و پرهیز از مشکلات ناشی از برخی راه حل های دشوار که پیاده سازی درگاه پرداخت اینترنتی را برای پذیرندگان اینترنتی دشوار می سازد، شرکت آسان پرداخت پرشین (پرشین سوئیچ) از فناوری وب سرویس (SOAP Web Services) برای برقراری خدمات پرداخت استفاده نموده است. با بکارگیری این فناوری، امکان پیاده سازی سرویس های عملیات پرداخت برای پذیرندگان اینترنتی با سهولت زیاد فراهم می شود و ضمن آن، استقلال این فناوری از بستر پیاده سازی و استفاده از استاندارد HTTP برای تبادل ارتباطات و بهره گیری از گواهینامه امنیتی SSL معتبر، امکان مجتمع سازی امن این خدمات را روی بستر استاندارد امکان پذیر می کند.

در سامانه درگاه پرداخت اینترنتی آسان پرداخت، از سوی آسان پرداخت متدهایی در اختیار پذیرنده قرار می گیرد که در سایت خود از آنها استفاده نماید. پذیرنده قادر خواهد بود این متدها را بصورت مستقیم از داخل کد وب سایت خود فراخوانی نماید.

انتقال اطلاعات بین وب سایت پذیرنده و سرویس درگاه پرداخت اینترنتی با پروتکل Simple Object Access Protocol (SOAP) خواهد بود. خود پروتکل SOAP برای دسته بندی و مدیریت داده ها از استاندارد XML استفاده می کند. در لایه Transport نیز انتقال داده ها بر عهده پروتکل HTTPS می باشد.

محدوده

هدف از ارائه این سند، معرفی سرویس های بهم پیوسته ایست که مجموعا خدمات درگاه پرداخت اینترنتی آسان پرداخت از منظر پذیرنده اینترنتی را شکل می دهند. محدوده این سند، شرح کلی عملیات لازم برای انجام عملیات پرداخت درخواستی از سوی پذیرنده روی درگاه پرداخت اینترنتی آسان پرداخت است و در کنار آن، تمامی متدهای مربوط به این عملیات شرح داده شده اند.

پیش نیازها

برای استفاده از این سرویس و اتصال به سرور پرداخت، لازم است در ابتدا IP وب سرور پذیرنده یا هر سرور دیگری که برای استفاده توسط وب سرور، عملیات مالی ساماندهی می کند و آدرس Domain صفحه ارجاع دهنده از وب سایت پذیرنده به درگاه پرداخت اینترنتی را به همراه درخواست استفاده از سرویس به آدرس ایمیل زیر ارسال فرمائید:

ipg@asanpardakht.ir

چنانچه این درخواست به همراه اطلاعات سرور ارسال نشود، آسان پرداخت اجازه دسترسی به درگاه پرداخت اینترنتی را نخواهد داد. در صورتی که درخواست مزبور از سوی آسان پرداخت مورد موافقت قرار بگیرد، پذیرنده از سوی آسان پرداخت پنج نوع اطلاعات زیر را دریافت خواهد نمود:

- کد پیکربندی پذیرنده یا Merchant Configuration ID
- نام کاربری یا Username
- رمز عبور یا Password
- کلید رمزنگاری
- IV رمزنگاری

لازم است اطمینان حاصل کنید که پورت های ۴۴۳ و ۸۰ روی سرور پذیرنده باز هستند و می توانند روی این دو پورت اطلاعات را ارسال و دریافت نمایند.

نحوه دسترسی به وب سرویس

جهت استفاده از وب سرویس مزبور باید آدرس های زیر را در داخل کد وب سایت خود در دسترس کنید:

۱	آدرس WSDL فراخوانی متدهای پرداخت و تکمیل پرداخت:	https://services.asanpardakht.net/paygate/merchantservices.asmx
۲	آدرس WSDL فراخوانی متد مشاهده آدرس کلاینت وب سرویس:	https://services.asanpardakht.net/utils/hostinfo.asmx

۳	آدرس WSDL متد مشاهده تاریخ و زمان فعلی سرور پرداخت:	https://services.asanpardakht.net/paygate/ servertime.asmx
۴	آدرسی که باید درخواست استفاده از درگاه پرداخت به آن POST شود:	https://asan.shaparak.ir
۵	آدرس سرویس استعلام نتیجه تراکنش با شماره فاکتور محلی	https://services.asanpardakht.net/paygate/statuswatch.asmx

جدول شماره ۱ – آدرس های سرویس درگاه پرداخت اینترنتی آسان پرداخت

همه گواهینامه های SSL شرکت آسان پرداخت پرشین برای خود درگاه پرداخت و برای وب سرویس ها معتبر است. چنانچه در کد خود برای نسخه های قبلی از روش Bypass نمودن گواهینامه Self-signed استفاده می کنید لازم است آن بخش از کد ها را حذف نمایید.

شرح متدهای بکار رفته در سرویس دهی درگاه پرداخت اینترنتی

RequestOperation

از این متد برای درخواست انجام تراکنش از درگاه پرداخت اینترنتی آسان پرداخت استفاده می شود. در صورتی که صحت اعتبار پذیرنده توسط آسان پرداخت تایید شود (بوسیله ارسال پارامترهای ورودی)، کد یکتایی برای او صادر و ارسال می شود. مقدار بازگشتی یک رشته است که از یک یا دو قسمت تشکیل شده است. کاراکتر Delimiter بین دو بخش نتیجه عملیات، کاراکتر کاما (,) می باشد.

الف) اگر عملیات موفق باشد، بخش اول آن عدد صفر، سپس یک کاراکتر کاما و سپس یک Hash Code به فراخواننده متد بازگردانده می شود که لازم است در مرحله بعد، پذیرنده آن Hash Code را در فرم صفحه وب مرتبط خود قرار داده و آن را به سمت آدرس ذکر شده در ردیف ۴، POST کند:

ب) چنانچه عملیات ناموفق باشد، تنها یک کد بازگردانده خواهد شد که کد مزبور نشان دهنده کد خطایی است که اتفاق افتاده است. تفسیر کد مزبور در ضمیمه الف این سند اورده شده است.

نحوه فراخوانی متد و روش ارسال پارامترها:

برای فراخوانی متد RequestOperation لازم است فایل WSDL زیر را به پروژه سایت خود بیفزایید یا آنکه چنانچه محیط برنامه سازی شما از این امکان برخوردار نیست به روش مقتضی فایل WSDL را در سایت خود آماده فراخوانی کنید.

نحوه فراخوانی متد RequestOperation به صورت زیر است:

```
string RequestOperation(int merchantConfigurationID, string encryptedRequest)
```

متد مزبور دو پارامتر دارد که نخستین آن، merchantConfigurationID، همان کد پیکربندی پذیرنده است که پیشتر در اختیار پذیرنده قرار داده شده است. پارامتر دوم یا همان encryptedRequest رشته ای است که باید به روشی که در ادامه می آید آماده شود.

روش آماده سازی رشته رمز شده پارامتر متد RequestOperation بدون درخواست تقسیم وجوه:

برای آماده سازی این رشته لازم است نخست رشته ای ایجاد کنید از ۹ پارامتر زیر که صرفا با یک کاراکتر کامای لاتین (,) از یکدیگر جدا شوند. توجه کنید که Delimiter تنها کاراکتر کاماست و استفاده از سایر کاراکترها همانند فاصله و غیره برای جداسازی پارامترها مجاز نیست.

این ۹ پارامتر می بایست صرفا با یک ترتیب مشخص در رشته قرار بگیرند و پذیرنده مجاز به تغییر محل نسبی قرارگیری پارامتری در درون رشته نمی باشد.

ترتیب قرار گیری پارامترها در درون رشته به شرح زیر است:

P1,P2,P3,P4,P5,P6,P7,P8,P9

نشانه	مفهوم	مثال
P1	کد نوع سرویس. برای انجام خرید کد ۱ را ارسال بفرمائید.	1
P2	نام کاربری که از سوی آسان پرداخت به پذیرنده اختصاص یافته است.	
P3	رمز عبوری که از سوی آسان پرداخت به پذیرنده اختصاص یافته است.	
P4	شناسه فاکتور. لازم است پذیرنده در هنگام ارسال درخواست خرید خود، شناسه منحصر به فرد و عددی را ارسال نماید. این شناسه می تواند هر عدد مثبت صحیحی باشد. این شناسه نمی تواند به هیچ وجه تکراری باشد.	

P5	مبلغ فاکتور به ریال بدون هیچ Separator ای می بایست ارسال شود. مبلغ مزبور می بایست در رنج حداکثر و حداقلی باشد که از نظر آسان پرداخت برای پذیرنده مجاز شناخته شده است و پذیرنده هنگام درخواستش برای فعال سازی درگاه پرداخت می بایست این حداقل و حداکثر را مشخص نماید.	
P6	تاریخ میلادی و زمان ارسال درخواست. این تاریخ و ساعت می بایست به فرمت YYYYMMDD HHMMSS ارسال شود. این زمان می بایست به صورت Synchronize شده با زمان سرور پرداخت آسان پرداخت ارسال شود و اختلاف زمان و تاریخ بیشتر از حداکثر یک روز مجاز نیست. برای سهولت کار پذیرندگان و بالاخص در شرایطی که پذیرندگان امکان تنظیم ساعت سروری که وب سایتشان را روی آن اجرا می کنند، متدی به نام GetPaymentServerTime تعبیه شده که در همین سند می توانید نحوه استفاده از آن را بیابید.	20120603 182231
P7	اطلاعات اضافی تراکنش را در این فیلد ارسال فرمائید. این اطلاعات حداکثر می بایست طولی برابر ۱۰۰ کاراکتر داشته باشند و در غیر اینصورت Truncate خواهند شد.	
P8	در این فیلد لازم است آدرس Callback ای که پاسخ درگاه پرداخت به آن صفحه ارسال خواهد شد را ارسال کنید. این آدرس می بایست با http یا https شروع شود و اکیدا توصیه می شود از شروع کردن URL ها با آدرس IP اجتناب بفرمائید. چنانچه پذیرنده آدرس پیش فرضی را برای Callback URL خود مشخص کرده باشد می تواند این فیلد را خالی بگذارد.	
P9	شناسه پرداخت. چنانچه عملیات شما از این ویژگی پشتیبانی نمی کند فیلد مزبور را 0 ارسال کنید.	

نکته مهم: چنانچه به اقتضا شرایط بخواهید فیلدی را خالی بگذارید لازم است در هر حال و هر شرایطی کاراکتر Delimiter را درج نمائید.

آماده سازی رشته در حالت تمایل به تقسیم وجوه

در صورتی که پذیرنده تمایل به تقسیم وجوه ناشی از تراکنش داشته باشد، به گونه ای که بخواهد بین حساب های مختلفی وجه تراکنش تقسیم شود، می تواند به صورت زیر درخواست انجام تراکنش را ارسال نماید:

P1,P2,…,P9, شبا ,بخشی از مبلغ تراکنش ,شبا ,بخشی از مبلغ تراکنش ,…

همه شماره های شبایی که پذیرنده در این درخواست عنوان می دارد می بایست قبلا برای پذیرنده، توسط آسان پرداخت و نزد شاپرک ثبت شده باشند. بخش های مبالغ اظهار شده نیز می بایست به ریال عنوان شوند و مجموع این مبالغ با فیلد P5 مطابقت داشته باشد.

تقسیم وجوه می تواند به حداکثر ۷ شبای تعریف شده توسط پذیرنده انجام بگیرد.

روش رمز سازی رشته ایجاد شده

برای رمز نگاری رشته بدست آمده از مراحل بالا و همچنین رشته های رمز شده ای که در متدهای دیگر درگاه پرداخت مورد نیاز هستند لازم است از الگوریتم AES با مشخصه های زیر استفاده شود:

- اندازه کلید و بلوک: ۲۵۶ بیت
- نوع Padding: PKCS7
- کلید: همان کلیدی که از سوی آسان پرداخت به پذیرنده تخصیص داده شده است.
- IV: همان IV ای که از سوی آسان پرداخت به پذیرنده تخصیص داده شده است.
- Encoding مورد استفاده: UTF-8

همچنین لازم به ذکر است پاسخ تراکنش ها نیز از سوی درگاه پرداخت به سمت سایت پذیرنده بصورت رمز نگاری شده و با همین روش به ایشان اطلاع داده می شوند. بنابراین لازم است پذیرنده هم متد Encryption و هم متد Decryption را در سمت خود با مشخصه های ذکر شده پیاده سازی نمایند.

در زیر نمونه کدهایی برای رمزنگاری به روش بالا در زبان های مختلف آورده شده است:

نمونه کد رمزنگاری به زبان C#:

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Security.Cryptography;
using System.IO;
```

```
namespace AESUtility
{
    public class AES
    {
        string AES_Key = string.Empty;
        string AES_IV = string.Empty;
        public AES(string AES_Key, string AES_IV)
        {
            this.AES_Key = AES_Key;
            this.AES_IV = AES_IV;
        }

        public bool Encrypt(String Input, out string encryptedString)
        {
            try
            {
                var aes = new RijndaelManaged();
                aes.KeySize = 256;
                aes.BlockSize = 256;
                aes.Padding = PaddingMode.PKCS7;
                aes.Key = Convert.FromBase64String(this.AES_Key);
                aes.IV = Convert.FromBase64String(this.AES_IV);

                var encrypt = aes.CreateEncryptor(aes.Key, aes.IV);
                byte[] xBuff = null;
                using (var ms = new MemoryStream())
                {
                    using (var cs = new CryptoStream(ms, encrypt, CryptoStreamMode.Write))
                    {
                        byte[] xXml = Encoding.UTF8.GetBytes(Input);
                        cs.Write(xXml, 0, xXml.Length);
                    }

                    xBuff = ms.ToArray();
                }

                encryptedString = Convert.ToBase64String(xBuff);
                return true;
            }
            catch (Exception ex)
            {
                encryptedString = string.Empty;
                return false;
            }
        }

        public bool Decrypt(String Input, out string decodedString)
        {
            try
            {
                RijndaelManaged aes = new RijndaelManaged();
                aes.KeySize = 256;
                aes.BlockSize = 256;
                aes.Mode = CipherMode.CBC;
                aes.Padding = PaddingMode.PKCS7;
                aes.Key = Convert.FromBase64String(this.AES_Key);
                aes.IV = Convert.FromBase64String(this.AES_IV);

                var decrypt = aes.CreateDecryptor();
                byte[] xBuff = null;
                using (var ms = new MemoryStream())
                {
                    using (var cs = new CryptoStream(ms, decrypt, CryptoStreamMode.Write))
                    {
                        byte[] xXml = Convert.FromBase64String(Input);
                        cs.Write(xXml, 0, xXml.Length);
                    }

                    xBuff = ms.ToArray();
                }

                decodedString = Encoding.UTF8.GetString(xBuff);
                return true;
```

```
            }
            catch (Exception ex)
            {
                decodedString = string.Empty;
                return false;
            }
        }

    }
}
```

نمونه کد رمزنگاری به زبان PHP:

```
<?php
// enable extension=php_mcrypt.dll AND extension=php_soap.dll on php.ini

date_default_timezone_set('Asia/Tehran');

$KEY = "Your KEY";
$IV = "Your IV";
$username = "user";
$password = "pass";

function addpadding($string, $blocksize = 32)
{
    $len = strlen($string);
    $pad = $blocksize - ($len % $blocksize);
    $string .= str_repeat(chr($pad), $pad);
    return $string;
}

function strippadding($string)
{
    $slast = ord(substr($string, -1));
    $slastc = chr($slast);
    $pcheck = substr($string, -$slast);
    if(preg_match("/$slastc{".$slast."}/", $string)){
        $string = substr($string, 0, strlen($string)-$slast);
        return $string;
    } else {
        return false;
    }
}

function encrypt($string = "")
{
global $KEY,$IV;
    $key = base64_decode($KEY);
    $iv = base64_decode($IV);
       return base64_encode(mcrypt_encrypt(MCRYPT_RIJNDAEL_256, $key, addpadding($string),
MCRYPT_MODE_CBC, $iv));
}

function decrypt($string = "")
{
global $KEY,$IV;
    $key = base64_decode($KEY);
    $iv = base64_decode($IV);
       $string = base64_decode($string);
       return strippadding(mcrypt_decrypt(MCRYPT_RIJNDAEL_256, $key, $string, MCRYPT_MODE_CBC,
$iv));
}
?>
```

نمونه کد رمزنگاری به زبان جاوا:

```
package ir.asanpardakht.javaclasses;

import java.nio.charset.Charset;
```

```
import org.bouncycastle.crypto.CipherParameters;
import org.bouncycastle.crypto.engines.RijndaelEngine;
import org.bouncycastle.crypto.modes.CBCBlockCipher;
import org.bouncycastle.crypto.paddings.PKCS7Padding;
import org.bouncycastle.crypto.paddings.PaddedBufferedBlockCipher;
import org.bouncycastle.crypto.params.KeyParameter;
import org.bouncycastle.crypto.params.ParametersWithIV;
import org.codehaus.plexus.util.Base64;

public final class StaticFunctions {

    private static final String ASAN_PARDAKHT_SECRET_KEY = "###############";
    private static final String ASAN_PARDAKHT_INITIALIZATION_VECTOR = "###############";
    private static final String CHARSET_NAME = "UTF-8";

    public static String asanPardakhtEncryption(String text) throws Exception {
        PaddedBufferedBlockCipher aes = new PaddedBufferedBlockCipher(new CBCBlockCipher(new
RijndaelEngine(256)), new PKCS7Padding());
        CipherParameters ivAndKey = new ParametersWithIV(new
KeyParameter(Base64.decodeBase64(ASAN_PARDAKHT_SECRET_KEY.getBytes(CHARSET_NAME))),
Base64.decodeBase64(ASAN_PARDAKHT_INITIALIZATION_VECTOR.getBytes(CHARSET_NAME)));
        aes.init(true, ivAndKey);

        int outputSize = aes.getOutputSize(text.length());
        byte[] encryptedBytes = new byte[outputSize];
        int length = aes.processBytes(text.getBytes(),
                0,
                text.length(),
                encryptedBytes,
                0);
        aes.doFinal(encryptedBytes, length);
        return new String(Base64.encodeBase64(encryptedBytes), CHARSET_NAME);
    }

    public static String asanPardakhtDecryption(String encryptedText) throws Exception {
        PaddedBufferedBlockCipher aes = new PaddedBufferedBlockCipher(new CBCBlockCipher(new
RijndaelEngine(256)), new PKCS7Padding());
        CipherParameters ivAndKey = new ParametersWithIV(new
KeyParameter(Base64.decodeBase64(ASAN_PARDAKHT_SECRET_KEY.getBytes(CHARSET_NAME))),
Base64.decodeBase64(ASAN_PARDAKHT_INITIALIZATION_VECTOR.getBytes(CHARSET_NAME)));
        aes.init(false, ivAndKey);

        byte[] decodedBytes = Base64.decodeBase64(encryptedText.getBytes(CHARSET_NAME));
        int outputSize = aes.getOutputSize(decodedBytes.length);
        byte[] decryptedBytes = new byte[outputSize];
        int length1 = aes.processBytes(decodedBytes,
                0,
                decodedBytes.length,
                decryptedBytes,
                0);
        int length2 = aes.doFinal(decryptedBytes, length1);

        return new String(decryptedBytes,
                0,
                length1 + length2,
                Charset.defaultCharset());
    }

}
```

شیوه ارجاع مشتری به درگاه پرداخت

برای ارجاع مشتری به درگاه پرداخت، لازم است Hidden Field ای با نام RefID را با توکن دریافت شده از متد RequestOperation پر کرده و به فرم را به آدرس گفته شده در ردیف چهارم جدول ۱ به روش POST ارسال نماید. با این کار مشتریِ پذیرنده به درگاه پرداخت وارد شده و آماده انجام تراکنش مالی می گردد.

برای راحتی کار می توانید از تابع زیر برای انجام عملیات ارجاع به درگاه پرداخت استفاده کنید. مبرهن است که این تابع را باید برای کاربرد خاص خود تغییر دهید تا متناسب با سناریوی مدنظرتان گردد:

```
<script language="javascript" type="text/javascript">
    function postRefId(refIdValue) {
        var form = document.createElement("form");
        form.setAttribute("method", "POST");
        form.setAttribute("action", "https://asan.shaparak.ir");
        form.setAttribute("target", "_self");
        var hiddenField = document.createElement("input");
        hiddenField.setAttribute("name", "RefId");
        hiddenField.setAttribute("value", refIdValue);
        form.appendChild(hiddenField);
        document.body.appendChild(form);
        form.submit();
        document.body.removeChild(form);
    }
</script>
```

چنانچه مایلید مشتریانتان از سرویس انتخاب کارت درگاه پرداخت نیز بهره ببرند، لازم است در کنار refIdValue از پارامتر mobileap نیز استفاده کنید که در بخش شیوه کار با سرویس انتخاب کارت شرح داده شده است

روش شروع پرداخت بدون دریافت توکن

پذیرنده می تواند بدون انجام روال های بخش قبل و بدون بهره گیری از متد RequestOperation و بدون نیاز به آنکه توکنی دریافت کند که به توسط آن وارد درگاه شود، به کمک روش زیر، بصورت مستقیم و بدون توکن، مشتری را به درگاه پرداخت اینترنتی ارجاع دهد. در این روش، کافیست پذیرنده پارامترهایی را عینا طبق جدول زیر با عنوان فیلدهای جداگانه به درگاه پرداخت POST کند. در حقیقت، بجای آنکه پارامتر RefID روش با توکن را به صفحه پرداخت ارسال کند، پارامترهای زیر هستند که به درگاه ارسال می شوند:

mconfigid	کد پیکربندی پذیرنده
ordermix	معادل رمز شده عبارت زیر است: localinvoiceid,amount برای ایجاد این مقدار، لازم است رشته ای شامل شماره فاکتور محلی و مبلغ را به ترتیب بالا ایجاد کرده و سپس آن را با کلید و IV در اختیار خود رمز کنید.
callbackurl	آدرسی که بعد از انجام پرداخت موفق/فشار دادن دکمه انصراف در صفحه پرداخت، مشتری بدان ارجاع داده خواهد شد.

نکته: در صورت استفاده از سرویس انتخاب کارت، لازم است پارامتر آن را نیز به شکلی که توضیح داده شده است ارسال کنید.

پس از اتمام پرداخت، پذیرنده می تواند به هر دو روش اول و دوم ذکر شده در این مستند برای اطلاع پذیرنده از وقوع پرداخت، از نتیجه پرداخت آگاهی یابد و روش پرداخت بدون توکن با هر دو روش آگاهی از نتیجه پرداخت سازگار است.

شیوه کار با سرویس انتخاب کارت

سرویس انتخاب کارت به پذیرنده امکان می دهد در هنگام شروع پرداخت، با ارسال شماره موبایل مشتری اش به درگاه پرداخت در کنار پارامترهای دیگر، به درگاه پرداخت امکان دهد که چنانچه پیشتر مشتری با همان شماره موبایل کارتی را نزد آسان پرداخت ذخیره نموده بود، آن کارت در معرض انتخابش قرار بگیرد و بجای تایپ کردن اطلاعات کارت (به استثنای رمز و CVV) به سادگی بتواند کارت را انتخاب بکند.

برای استفاده از این خدمت، کافیست Hidden Field دیگری با نام mobileap و با مقدار شماره موبایل به فرمت (رشته ای) 09XXXXXXXXX در کنار پارامتر RefID گفته شده در بخش شیوه ارجاع مشتری به درگاه پرداخت ارسال کنید.

در حقیقت در صورت بهره گیری از مثال بخش شیوه ارجاع مشتری به درگاه پرداخت ،تابع انتقال مشتری به درگاه شبیه زیر خواهد بود:

```
<script language="javascript" type="text/javascript">
    function postRefId(refIdValue, mobileNumber) {
        var form = document.createElement("form");
        form.setAttribute("method", "POST");
        form.setAttribute("action", "https://asan.shaparak.ir");
        form.setAttribute("target", "_self");
        var hiddenField = document.createElement("input");
        hiddenField.setAttribute("name", "RefId");
        hiddenField.setAttribute("value", refIdValue);
        form.appendChild(hiddenField);
        var hiddenFieldMobile = document.createElement("input");
        hiddenFieldMobile.setAttribute("name", "mobileap");
        hiddenFieldMobile.setAttribute("value", mobileNumber);
        form.appendChild(hiddenFieldMobile);
        document.body.appendChild(form);
        form.submit();
        document.body.removeChild(form);
    }
</script>
```

این سرویس هم در پرداخت با توکن و هم در پرداخت بدون توکن قابل استفاده است.

این سرویس در هنگام پرداخت توسط گوشی های هوشمند نیز به راحتی قابل استفاده است. دسترسی به این سرویس بطور پیش فرض برای پذیرندگان برقرار نیست و لازم است پذیرندگان متقاضی، برای فعال شدن این سرویس، درخواست و دلایل خود را با آسان پرداخت در میان بگذارند.

اطلاع سایت پذیرنده از وقوع پرداخت

روش اول – گشودن رشته رمزشده حاوی نتیجه تراکنش

در بازگشت به صفحه پذیرنده ، یک عبارت رمز شده (متشکل از ۸ مقدار) بصورت POST بازگردانده می‌شود که عبارت بازگردانده شده با نام ReturningParams بوده و رشته ای است به شکل زیر (از چپ به راست):

```
Amount,SaleOrderId,RefId,ResCode,MessageText,PayGateTranID,RRN,LastFourDigitOfPAN
```

عنوان و شرح هر یک از فیلدهای رشته بالا (که با کاراکتر کاما از هم تفکیک شده اند) بصورت زیر است:

- Amount: مبلغ تراکنش به ریال.
- SaleOrderId: شماره فاکتور تراکنش که توسط پذیرنده تولید شده است.
- RefId: مقداری که در ابتدای تراکنش تولید و توسط پذیرنده به صفحه درگاه پرداخت پست شده است.
- ResCode: نتیجه تراکنش که در صورت 0 یا 00 بودن، تراکنش موفق و در غیر اینصورت کد خطای رخ داده برگردانده می‌شود.
- MessageText: پیغام ارسالی ناشی از نتیجه تراکنش که میتواند موفق بودن تراکنش و یا شرح خطای رخداده باشد. در این فیلد همچنین در حال حاضر، شش رقم ابتدای شماره کارت انجام دهنده تراکنش بازگردانده می شود.
- PayGateTranID: کد پیگیری تراکنش (اختصاصی آسان پرداخت پرشین)
- RRN: شناسه مرجع بانکی تراکنش
- LastFourDigitsOfPAN: چهار رقم انتهایی شماره کارت انجام دهنده تراکنش

پس از دریافت پارامترهای فوق ابتدا باید نتیجه تراکنش (ResCode) چک شود که در صورتیکه 0 یا 00 بود ، کار ادامه یابد و در غیراینصورت تراکنش ناموفق بوده و وضعیت آن مختومه می باشد.

توجه داشته باشید که برای رمزگشایی از رشته مزبور می بایست از همان کلید و Vector ای استفاده شود که در اختیار شما قرار گرفته است. عینا الگوریتم رمزگشایی AES با پارامترهایی است که پیشتر شرح آنها داده شد.

روش دوم – استعلام نتیجه تراکنش از طریق شماره فاکتور محلی

دریافت نتیجه تراکنش به این روش مستلزم آن است که در لحظه فراخوانی صفحه Callback اعلامی شما، بدانید که ورود به این صفحه برای کدام شماره فاکتور محلی (LocalInvoiceID در RequestOperation) صورت گرفته است. به عنوان مثال می توانید در مرحله نخست و هنگام اعلام Callback URL، شماره فاکتور محلی خود را در URL جا بدهید (داخل Query String) تا در هنگام بازگشت مشتری پس از پرداخت به صفحه Callback متوجه شوید که ورود به صفحه برای کدام فاکتور بوده است.

زمانی که دانستید ورود به ازای کدام شماره فاکتورتان بوده است، می توانید وب سرویس CheckTransactionResult در آدرس زیر را فراخوانی کنید تا اطلاعات تراکنش موفق رخ داده به ازای شماره فاکتور خود را دریافت کنید:

https://services.asanpardakht.net/paygate/statuswatch.asmx

در خصوص وب سرویس مزبور موارد زیر حائز اهمیت است:

۱- پاسخ وب سرویس مزبور به فرمت JSON به شما باز می گردد.

۲- تنها تا یک ساعت از لحظه انجام تراکنش موفق (اگر تراکنش وجود داشته باشد) وب سرویس مزبور به شما اطلاعات وضعیت تراکنش را باز می گرداند و پس از آن نتیجه ای را به شما باز نخواهد گرداند.

۳- با دریافت نتیجه لازم است در ابتدا فیلد Result نتیجه را چک کنید. چنانچه نتیجه بازگردانده شده نشاندهنده موفقیت عملیات بود، فیلدهای دیگر قابل بهره برداری اند و در غیر اینصورت فیلد دیگری وجود نخواهد داشت. برای مشاهده کدهای نشان دهنده موفقیت یا عدم موفقیت به جدول متناظر با وب سرویس مزبور در ضمیمه الف مراجعه بفرمایید.

۴- در صورت یافته شدن موفقیت آمیز اطلاعات، جزئیات مربوط به یک و تنها یک تراکنش موفق به شما بازگردانده خواهد شد. امکان وجود دو یا تعداد بیشتری تراکنش موفق به ازای یک شماره فاکتور محلی وجود ندارد.

۵- این وب سرویس تنها اطلاعات مربوط به تراکنش موفق را باز می گرداند. در صورت نبود تراکنشی موفق، انصراف یا بستن پنجره مرورگر توسط مشتری، شما با وضعیت *تراکنشی یافت نشد* مواجه خواهید شد.

۶- چنانچه نتیجه ردیف ۳ بیانگر بازیابی موفقیت آمیز اطلاعات مربوط به یک تراکنش موفق باشد، از تراکنش مزبور، حداقل اطلاعات زیر را بازخواهد گرداند (ممکن است برای برخی پذیرندگان اطلاعات بیشتری بازگردانده شود. با این حال حداقل اطلاعات مزبور برای همه پذیرندگان در دسترس خواهد بود):

Result	نتیجه تراکنش طبق شرح بند ۳
CardNumber	شماره کارت ماسک شده (تنها ۶ رقم ابتدا و ۴ رقم آخر قابل دسترسی اند)
RRN	شماره پیگیری شبکه بانکی
RefID	توکنی که در هنگام ارجاع مشتری از سایت پذیرنده به صفحه پرداخت، از سوی پذیرنده به صفحه درگاه POST شده است.
Amount	مبلغ تراکنش انجام شده
PayGateTranID	شماره تراکنش سوئیچ
SalesOrderID	شماره فاکتور محلی اعلام شده از سوی پذیرنده

مقادیر و نوع پارامترهای ورودی عینا همانند وب سرویس مورد بهره برداری برای بررسی وضعیت تراکنش (متد TrxStatusFromLocalInvoiceID) است. توجه بفرمایید که متد حاضر را تنها حداکثر یک ساعت بعد انجام تراکنش می توان فراخوانی کرد ولی متد TrxStatusFromLocalInvoiceID هر زمان قابل فراخوانی است.

نکته مهم: چه از روش اول از نتیجه پرداخت مطلع شوید و چه از روش دوم، در هر دو حالت ادامه فرآیند تکمیل خرید یکسان است و در سطر زیر بطور مبسوط شرح داده شده است.

متدهای پس از انجام پرداخت (و اطلاع سایت پذیرنده از وقوع پرداخت)

چنانچه پرداخت روی درگاه پرداخت صورت گرفته باشد، نتیجه پرداخت بر مبنای جدول ضمیمه ب همین سند به پذیرنده اعلام می شود. چنانچه درگاه پرداخت کدی غیر از صفر را به پذیرنده اعلام کند، به معنای ناموفق بودن عملیات پرداخت است و پذیرنده نیاز نیست کار خاصی را در رابطه با درگاه پرداخت انجام دهد. در صورتی که

درگاه پرداخت نتیجه صفر را به پذیرنده باز گردانده باشد، امکان استفاده از متدهای Post Payment درگاه پرداخت برای نهایی کردن وضعیت تراکنش وجود خواهد داشت.

نکته: کد نتیجه ۹۱۱ به معنای انصراف کاربر از نتیجه تراکنش می باشد (کاربر دکمه انصراف را در صفحه پرداخت فشار داده است).

دو مسیر برای نهایی کردن وضعیت تراکنش مالی موفق وجود دارند:

مسیر الف: Verify ← Reconciliation

مسیر ب: Reverse

چنانچه پذیرنده بخواهد تراکنش مالی منجر به تسویه با وی شود لازم است مسیر الف و در صورتی که بخواهد مبلغ تراکنش را به حساب خریدار برگشت بزند، مسیر ب را می بایست انتخاب کند.

هر سه متد مورد نیاز برای عملیات های Verification، Reconciliation و Reversal به پارامترهای مشابهی فراخوانی می شوند که نخست شرح پارامترهای مزبور آورده می شوند:

نام پارامتر	شرح
`merchantConfigurationID`	کد پیکربندی پذیرنده
`encryptedCredentials`	نام کاربری کاراکتر کاما رمز عبور که جملگی به روش گفته شده و با کلید و InitVector ارائه شده به پذیرنده می بایست رمز شده باشند.
`payGateTranID`	شماره پیگیری تراکنش اصلی که پس از انجام عملیات پرداخت توسط درگاه پرداخت به اطلاع پذیرنده رسانده شده است.

فایل WSDL همه متدهای Post Payment در ردیف ۱ جدول ۱ قرار دارند:

بنابراین لازم است فایل WSDL مزبور را به پروژه سایت خود بیفزایید یا آنکه چنانچه محیط برنامه سازی شما از این امکان برخوردار نیست به روش مقتضی این فایل WSDL را در سایت خود آماده فراخوانی کنید.

RequestVerification

پس از اطلاع یافتن پذیرنده از موفق بودن تراکنش مالی در درگاه پرداخت، لازم است پذیرنده بلافاصله این وب سرویس متد را فراخوانی نماید.

نحوه فراخوانی متد به صورت زیر است:

```
int RequestVerification(merchantConfigurationID, encryptedCredentials, payGateTranID)
```

که شرح آنها پیشتر آورده شد.

پاسخ این عملیات یک عدد است و می تواند یکی از مقادیر ذکر شده در ضمیمه الف، بخش *کد و تفسیر خطاهای مربوط به متد RequestVerification* باشد.

موارد زیر در رابطه با فراخوانی این متد حائز اهمیت هستند:

۱- یک تراکنش را نمی تواند بیش از یکبار Verify نمود.
۲- تراکنشی که برایش درخواست تسویه شده را نمی توان Verify نمود.
۳- تراکنشی که برایش درخواست بازگشت مبلغ تراکنش شده را نمی توان Verify نمود.
۴- تراکنش مالی غیرموفق را نمی توان Verify نمود.
۵- Verify تراکنش به معنای گام نخست برای درخواست تسویه آن است. چنانچه این متد را فراخوانی ننمائید امکان درخواست برای تسویه تراکنش را نخواهید داشت.
۶- عدم ارسال درخواست Verification تا ۳۰ دقیقه بعد از انجام تراکنش (به وقت سرور پرداخت) موجب خواهد شد تا بصورت خودکار تراکنش مزبور Reverse شود و مبلغ آن به حساب مشتری برگشت داده شود.
۷- همانگونه که از جدول کدهای نتیجه فراخوانی این متد پیداست، نتیجه ۵۰۰ به معنای موفقیت آمیز بودن عملیات است.

RequestReconciliation

پس از آنکه برای تراکنش مالی موفق توسط پذیرنده درخواست Verify شد، لازم است پذیرنده متد RequestReconciliation را فراخوانی نماید تا تراکنش در سیکل تسویه با پذیرنده قرار بگیرد.

اگر متد RequestReconciliation را فراخوانی نکنید، درخواست تسویه بعد از ۱۲ ساعت به سمت شاپرک ارسال می شود و طبعا واریز با تاخیر طولانی صورت خواهد گرفت.

نحوه فراخوانی این متد به صورت زیر است:

```
int RequestReconciliation(merchantConfigurationID, encryptedCredentials, payGateTranID)
```

پاسخ این عملیات یک عدد است و می تواند یکی از مقادیر ذکر شده در ضمیمه الف، بخش *کد و تفسیر خطاهای مربوط به متد RequestReconciliation* باشد.

در صورت موفق بودن نتیجه عمل که با کد پاسخ ۶۰۰ عینیت می یابد، پذیرنده می تواند اطمینان داشته باشد که بر مبنای سیکل تسویه سامانه شاپرک، تسویه حساب با پذیرنده صورت خواهد پذیرفت.

موارد زیر در رابطه با فراخوانی این متد حائز اهمیت هستند:

۱- یک تراکنش را نمی تواند بیش از یکبار Request for Reconciliation نمود.
۲- برای تراکنشی که Verify نشده نمی توان درخواست تسویه نمود.
۳- برای تراکنشی که برایش درخواست بازگشت مبلغ تراکنش شده نمی توان درخواست تسویه نمود.
۴- برای تراکنش مالی غیرموفق نمی توان درخواست تسویه نمود.
۵- همانگونه که از جدول کدهای نتیجه فراخوانی این متد پیداست، نتیجه ۶۰۰ به معنای موفقیت آمیز بودن عملیات است.

RequestReversal

چنانچه پذیرنده بخواهد مبلغ تراکنش مالی موفق را به حساب پذیرنده بازگرداند، به شرط آنکه قبلا تراکنش مزبور را Verify ننموده باشد، لازم است از متد RequestReversal استفاده کند. با فراخوانی این متد، درخواست پذیرنده برای پردازش و عودت وجه در سیکل پردازش سرور پرداخت قرار می گیرد.

نحوه فراخوانی این متد به صورت زیر است:

```
int RequestReversal(merchantConfigurationID, encryptedCredentials, payGateTranID)
```

پاسخ این عملیات یک عدد است و می تواند یکی از مقادیر ذکر شده در ضمیمه الف، بخش *کد و تفسیر خطاهای مربوط به متد RequestReversal* باشد.

در صورت موفق بودن نتیجه عمل که با کد پاسخ ۷۰۰ عینیت می یابد، پذیرنده می تواند اطمینان داشته باشد که بر مبنای سیکل تسویه سامانه شاپرک، مبلغ وجه تراکنش مالی موفق مزبور به خریدار عودت داده خواهد شد.

در رابطه با درخواست عودت وجه موارد زیر قابل ذکرند:

۱- تراکنش Verify شده را نمی توان Reverse نمود.
۲- تراکنش هایی که تا ۳۰ دقیقه پس از انجام تراکنش Verify نشده باشند، بصورت خودکار Reverse خواهند شد.
۳- برای تراکنش می توان تنها یکبار در صورت احراز شروط لازمش درخواست Reversal فرستاد.

CancelVerification

پذیرنده می تواند تراکنشی که Verify شده ولی برایش RequestReconciliate ننموده را در فاصله حداکثر ۳ ساعت پس از Verify نمودنش، کنسل کند.

نحوه فراخوانی متد به صورت زیر است:

```
int CancelVerification(merchantConfigurationID, encryptedCredentials, payGateTranID)
```

که شرح پارامترها همانند سایر متدهای تکمیلی (همانند RequestVerification) است.

پاسخ این عملیات یک عدد است و می تواند یکی از مقادیر ذکر شده در ضمیمه الف، بخش *کد و تفسیر خطاهای مربوط به متد CancelVerification* باشد.

موارد زیر در رابطه با فراخوانی این متد حائز اهمیت هستند:

۱- یک تراکنش را نمی تواند بیش از یکبار CancelVerify نمود.
۲- تراکنشی که برایش درخواست تسویه شده را نمی توان CancelVerify نمود.
۳- تراکنشی که برایش درخواست بازگشت مبلغ تراکنش شده را نمی توان CancelVerify نمود.
۴- تراکنش مالی غیرموفق را نمی توان CancelVerify نمود.

۵- همانگونه که از جدول کدهای نتیجه فراخوانی این متد پیداست، نتیجه ۵۰۰ به معنای موفقیت آمیز بودن عملیات است.

بررسی وضعیت تراکنش

متد TrxStatusFromLocalInvoiceID به شما امکان بررسی وضعیت تراکنش خود را بر اساس شماره فاکتور محلی خود (LocalInvoiceID) می دهد. نحوه فراخوانی این متد به صورت زیر است:

```
TransactionDataPackage TrxStatusFromLocalInvoiceID(int merchantConfigurationID, string encryptedCredintials, long localInvoiceID)
```

پارامتر encryptedCredintials به صورت *نام کاربری کاراکتر کاما رمز عبور که جملگی به روش گفته شده و با کلید و InitVector ارائه شده به پذیرنده می بایست رمز شده باشند* ساخته و ارسال می شود (مشابه متدهای RequestReversal و ...).

پاسخ این متد آبجکتی از کلاس TransactionDataPackage است که این آبجکت دارای Property های زیر می باشد:

`int fetchResult`	کدی که نتیجه عملیات (و نه تراکنش) را باز می گرداند. تفسیر کدهای نتیجه در ضمیمه الف آورده شده اند.
`DateTime reportDate`	تاریخ استعلام
`List<TransactionStatus> transactionList`	لیستی از تراکنش ها به ازای localInvoiceID ارسال شده. هر تراکنش آبجکتی از کلاس TransactionStatus است.

ویژگی های کلاس TransactionStatus

در زیر تمامی ویژگی های کلاس مزبور به همراه نوع داده ای هر یک آورده شده است:

نام Property	شرح
`int EMerchantConfigID`	کد پیکربندی پذیرنده
`long LocalInvoiceID`	شماره فاکتور محل
`DateTime OperationRequestDate`	تاریخ دریافت درخواست عملیات
`DateTime MerchantLocalDate`	تاریخ محلی ارسالی پذیرنده

long RequestedAmountInRials	مبلغ درخواستی پذیرنده برای انجام تراکنش
string AdditionalData	اطلاعات اضافی ارسالی از سوی پذیرنده
int ServiceTypeID	کد نوع سرویس
string CallbackURL	آدرس کال بک (صفحه بازگشتی اظهار شده در درخواست)
string PayerID	شناسه پرداخت
string Token	توکن دریافتی از متد RequestOperation
long PayGateTranID	شماره داخلی تراکنش
DateTime TransactionDate	تاریخ انجام تراکنش
string CardNumber	شماره کارت انجام دهنده تراکنش (برخی ارقام ماسک شده اند)
int Stan	استن تراکنش
long ActualTrxAmount	مبلغ تراکنش ارسالی
string BillID	شناسه قبض (صرفا در تراکنش های پرداخت قبض)
string PaymentID	شناسه پرداخت (صرفا در تراکنش های پرداخت قبض)
string TrxResultCode	کد نتیجه تراکنش
bool MainTrxWasSuccessful	فلگ مشخص کننده موفق بودن تراکنش اصلی
string RRN	RRN تراکنش
bool IsVerified	آیا قبلا Verify شده است
DateTime? VerifyRequestDate	تاریخ ارسال Verify
bool ReconcilationSucceeded	آیا تسویه موفق بوده است
DateTime? ReconcilationRequestDate	تاریخ درخواست تسویه
bool IsReconcilationRequestedAutomatically	درخواست تسویه بصورت خودکار و به نیابت از پذیرنده ارسال شده است؟
DateTime? ReconcilationTryDate	تاریخ آخرین ارسال تسویه به سمت شاپرک
bool ReversalSucceeded	ریورس موفق بوده است؟
DateTime? ReversalRequestDate	تاریخ درخواست عودت وجه
bool IsReversalRequestedAutomatically	آیا بدون درخواست پذیرنده تراکنش ریورس شده است؟ (معمولا در شرایط عدم دریافت درخواست Verify ظرف سی دقیقه از انجام تراکنش)

DateTime? ReversalTryDate	تاریخ آخرین ارسال عودت وجه به سمت شاپرک

ضمیمه الف- شرح کدهای خطای بازگشتی درگاه پرداخت

کد و تفسیر خطاهای مربوط به متد OperationRequest

301	پیکربندی پذیرنده اینترنتی نامعتبر است
302	کلیدهای رمزنگاری نامعتبر هستند
303	رمزنگاری نامعتبر است
304	تعداد عناصر درخواست نامعتبر است
305	نام کاربری یا رمز عبور پذیرنده نامعتبر است
306	با آسان پرداخت تماس بگیرید
307	سرور پذیرنده نامعتبر است
308	شناسه فاکتور می بایست صرفا عدد باشد
309	مبلغ فاکتور نادرست ارسال شده است
310	طول فیلد تاریخ و زمان نامعتبر است
311	فرمت تاریخ و زمان ارسالی پذیرنده نامعتبر است
312	نوع سرویس نامعتبر است
313	شناسه پرداخت کننده نامعتبر است
315	فرمت توصیف شیوه تسهیم شبا نامعتبر است
316	شیوه تقسیم وجوه با مبلغ کل تراکنش همخوانی ندارد
317	شبا متعلق به پذیرنده نیست
318	هیچ شبایی برای پذیرنده موجود نیست
319	خطای داخلی. دوباره درخواست ارسال شود
320	شبای تکراری در رشته درخواست ارسال شده است
321	فرمت آدرس کال بک نامعتبر است
322	آدرس کال بک اعلامی غیرمجاز است
-100	تاریخ ارسالی محلی پذیرنده نامعتبر است
-103	مبلغ فاکتور برای پیکربندی فعلی پذیرنده معتبر نمی باشد
-106	سرویس وجود ندارد یا برای پذیرنده فعال نیست
-109	هیچ آدرس کال بکی برای درخواست پیکربندی نشده است
-112	شماره فاکتور نامعتبر یا تکراری است **فیلد P4 را بطور صحیح و غیر تکراری ارسال کنید تا مشکل رفع شود**
-115	پذیرنده فعال نیست یا پیکربندی پذیرنده غیرمعتبر است

کد و تفسیر خطاهای مربوط به متد RequestVerification (درخواست بازبینی)

500	بازبینی تراکنش با موفقیت انجام شد
501	پردازش هنوز انجام نشده است
502	وضعیت تراکنش نامشخص است

503	تراکنش اصلی ناموفق بوده است **چنانچه تراکنش اصلی ناموفق باشد نمی توانید برای آن این درخواست را ارسال کنید**
504	قبلا درخواست بازبینی برای این تراکنش داده شده است
505	قبلا درخواست تسویه برای این تراکنش ارسال شده است
506	قبلا درخواست بازگشت برای این تراکنش ارسال شده است
507	تراکنش در لیست تسویه قرار دارد
508	تراکنش در لیست بازگشت قرار دارد
509	امکان انجام عملیات به سبب وجود مشکل داخلی وجود ندارد
510	هویت درخواست کننده عملیات نامعتبر است
511	قبلا درخواست کنسلی بازبینی برای این تراکنش شده است

کد و تفسیر خطاهای مربوط به متد RequestReconcilation

600	درخواست تسویه تراکنش با موفقیت ارسال شد
601	پردازش هنوز انجام نشده است
602	وضعیت تراکنش نامشخص است
603	تراکنش اصلی ناموفق بوده است **چنانچه تراکنش اصلی ناموفق باشد نمی توانید برای آن این درخواست را ارسال کنید**
604	تراکنش بازبینی نشده است
605	قبلا درخواست بازگشت برای این تراکنش ارسال شده است
606	قبلا درخواست تسویه برای این تراکنش ارسال شده است
607	امکان انجام عملیات به سبب وجود مشکل داخلی وجود ندارد
608	تراکنش در لیست منتظر بازگشت ها وجود دارد
609	تراکنش در لیست منتظر تسویه ها وجود دارد
610	هویت درخواست کننده عملیات نامعتبر است
611	قبلا درخواست کنسلی بازبینی برای این تراکنش شده است

کد و تفسیر خطاهای مربوط به متد RequestReversal

700	درخواست بازگشت تراکنش با موفقیت ارسال شد
701	پردازش هنوز انجام نشده است
702	وضعیت تراکنش نامشخص است
703	تراکنش اصلی ناموفق بوده است
704	امکان بازگشت یک تراکنش بازبینی شده وجود ندارد
705	قبلا درخواست بازگشت تراکنش برای این تراکنش ارسال شده است
706	قبلا درخواست تسویه برای این تراکنش ارسال شده است
707	امکان انجام عملیات به سبب وجود مشکل داخلی وجود ندارد
708	تراکنش در لیست منتظر بازگشت ها وجود دارد

709	تراکنش در لیست منتظر تسویه ها وجود دارد
710	هویت درخواست کننده عملیات نامعتبر است

کد و تفسیر خطاهای مربوط به متد CancelVerification

800	درخواست لغو بازبینی تراکنش با موفقیت ارسال شد
801	پردازش هنوز انجام نشده است
802	وضعیت تراکنش نامشخص است
803	تراکنش اصلی ناموفق بوده است **چنانچه تراکنش اصلی ناموفق باشد نمی توانید برای آن این درخواست را ارسال کنید**
804	تراکنش بازبینی نشده است
805	قبلا درخواست تسویه برای این تراکنش ارسال شده است
806	قبلا درخواست بازگشت برای این تراکنش ارسال شده است
807	تراکنش در لیست منتظر تسویه ها وجود دارد
808	تراکنش در لیست منتظر بازگشت ها وجود دارد
809	امکان انجام عملیات به سبب وجود مشکل داخلی وجود ندارد
810	هویت درخواست کننده عملیات نامعتبر است
811	این امکان برای تراکنش های پرداخت قبض وجود ندارد
812	جزئیات بازبینی یافت نشد
813	بازبینی قبلا لغو شده است
814	امکان درخواست بعد از گذشت بازه زمانی مجاز پس از `Verification` وجود ندارد.

کد و تفسیر خطاهای مربوط به متد TrxStatusFromLocalInvoiceID

400	موفق
401	حالت اولیه (مقدار اولیه در شرایط `Serialize/Deserialize`)
402	هویت درخواست کننده نامعتبر است
403	تراکنشی یافت نشد
404	خطا در پردازش

کد و تفسیر خطاهای مربوط به متد CheckTransactionResult

1100	موفق
1101	هویت درخواست کننده نامعتبر است
1102	خطا در پردازش
1103	تراکنشی یافت نشد

ضمیمه ب- شرح کدهای نتیجه انجام تراکنش مالی

ردیف	کد نتیجه	مفهوم کد
۱	0	تراکنش با موفقیت انجام شد
۲	1	صادرکننده کارت از انجام تراکنش صرف نظر کرد.
۳	2	عملیات تاییدیه این تراکنش قبلاً با موفقیت صورت پذیرفته است.
۴	3	پذیرنده فروشگاهی نامعتبر می باشد
۵	4	کارت توسط دستگاه ضبط شود.
۶	5	به تراکنش رسیدگی نشد.
۷	6	بروز خطا.
۸	7	به دلیل شرایط خاص کارت توسط دستگاه ضبط شود
۹	8	با تشخیص هویت دارنده ی کارت، تراکنش موفق می باشد.
۱۰	12	تراکنش نامعتبر است.
۱۱	13	مبلغ تراکنش اصلاحیه نادرست است.
۱۲	14	شماره کارت ارسالی نامعتبر است.(وجود ندارد)
۱۳	15	صادر کننده ی کارت نامعتبر است.(وجود ندارد)
۱۴	16	تراکنش مورد تایید است و اطلاعات شیار سوم کارت به روز رسانی شود
۱۵	19	تراکنش مجدداً ارسال شود.
۱۶	23	کارمزد ارسالی پذیرنده غیر قابل قبول است.
۱۷	25	تراکنش اصلی یافت نشد.
۱۸	30	قالب پیام دارای اشکال است.
۱۹	31	پذیرنده توسط سوئیچ پشتیبانی نمی شود.
۲۰	33	تاریخ انقضای کارت سپری شده است
۲۱	34	تراکنش اصلی با موفقیت انجام نپذیرفته است.
۲۲	36	کارت محدود شده است کارت توسط دستگاه ضبط شود.
۲۳	38	تعداد دفعات ورود رمز غلط بیش از حد مجاز است
۲۴	39	کارت حساب اعتباری ندارد.
۲۵	40	عملیات درخواستی پشتیبانی نمی گردد.
۲۶	41	کارت مفقودی می باشد. کارت توسط دستگاه ضبط شود.
۲۷	42	کارت حساب عمومی ندارد.
۲۸	43	کارت مسروقه می باشد. کارت توسط دستگاه ضبط شود.

۲۹	44	کارت حساب سرمایه گذاری ندارد.
۳۰	51	موجودی کافی نمی باشد.
۳۱	52	کارت حساب جاری ندارد.
۳۲	53	کارت حساب قرض الحسنه ندارد.
۳۳	54	تاریخ انقضای کارت سپری شده است.
۳۴	55	رمز کارت نامعتبر است.
۳۵	56	کارت نامعتبر است.
۳۶	57	بانک شما این تراکنش را پشتیبانی نمیکند
۳۷	58	انجام تراکنش مربوطه توسط پایانه ی انجام دهنده مجاز نمی باشد.
۳۸	59	کارت مظنون به تقلب است.
۳۹	61	مبلغ تراکنش بیش از حد مجاز می باشد.
۴۰	62	کارت محدود شده است.
۴۱	63	تمهیدات امنیتی نقض گردیده است.
۴۲	64	مبلغ تراکنش اصلی نامعتبر است.(تراکنش مالی اصلی با این مبلغ نمی باشد).
۴۳	65	تعداد درخواست تراکنش بیش از حد مجاز می باشد.
۴۴	67	کارت توسط دستگاه ضبط شود.
۴۵	75	تعداد دفعات ورود رمز غلط بیش از حد مجاز است.
۴۶	77	روز مالی تراکنش نا معتبر است.
۴۷	78	کارت فعال نیست.
۴۸	79	حساب متصل به کارت نامعتبر است یا دارای اشکال است.
۴۹	80	تراکنش موفق عمل نکرده است.
۵۰	84	بانک صادر کننده کارت پاسخ نمیدهد
۵۱	86	موسسه ارسال کننده شاپرک یا مقصد تراکنش در حالت Sign off است.
۵۲	90	بانک صادرکننده کارت درحال انجام عملیات پایان روز میباشد
۵۳	91	بانک صادر کننده کارت پاسخ نمیدهد
۵۴	92	مسیری برای ارسال تراکنش به مقصد یافت نشد. (موسسه های اعلامی معتبر نیستند)
۵۵	93	تراکنش با موفقیت انجام نشد. (کمبود منابع و نقض موارد قانونی)
۵۶	94	ارسال تراکنش تکراری.
۵۷	96	بروز خطای سیستمی در انجام تراکنش.
۵۸	97	فرایند تغییر کلید برای صادر کننده یا پذیرنده در حال انجام است.

ضمیمه ج- راهنمای گام به گام انجام تراکنش با درگاه پرداخت

بصورت خلاصه، برای انجام تراکنش با درگاه پرداخت باید مراحل زیر را طی نمائید:

۱- در ابتدا لازم است فروشگاه حقیقی یا حقوقی مایل به دریافت درگاه پرداخت اینترنتی شرکت آسان پرداخت پرشین، درخواست اخذ پذیرندگی اینترنتی از شرکت نماید. با این درخواست، از سوی شرکت با ایشان تماس گرفته خواهد شد و مراحل حقوقی لازم به اطلاع ایشان رسانده شده و فرم های لازم جهت درخواست در اختیار ایشان قرار می گیرد.

۲- در صورت موافقت شرکت و پس از قطعی شدن توافق شرکت با پذیرندگی فروشگاه، لازم است سه نوع اطلاعات زیر در اختیار شرکت قرار بگیرد: نام فروشگاه، حداکثر سه آدرس آی پی و شماره شبایی که با آن تسویه تراکنش ها صورت خواهد گرفت.

۳- پس از دریافت اطلاعات بند دو، مجموعه اطلاعات لازم برای اتصال به درگاه پرداخت در اختیار پذیرنده قرار خواهد گرفت که از آن میان می توان به نام کاربری، رمز عبور، کلیدهای رمزنگاری و سایر موارد اشاره نمود.

۴- حال لازم است پذیرنده متد OperationRequest و متدهای پس از انجام پرداخت را در سمت خود پیاده سازی کند. بصورت گام به گام، این رویه به صورت زیر خواهد بود:

a. طبق دستورالعمل گفته شده، متد OperationRequest را پیاده سازی کنید. تست های لازم را انجام دهید تا اطمینان حاصل کنید که پاسخ موفق را از این متد دریافت می کنید.

b. بخش دوم پاسخ صحیح دریافتی از متد OperationRequest را طبق دستورالعمل گفته شده در همین مستند به آدرس ردیف ۴ جدول ۱ ارسال کنید.

c. صفحه Callback را طوری برنامه ریزی کنید که بتواند پاسخ درگاه پرداخت را دریافت کند و بر مبنای نوع پاسخ در مورد گام بعدی تصمیم گیری کند.

d. در صورتی که پاسخ درگاه پرداخت حاکی از موفقیت آمیز بودن تراکنش بود، لازم است طبق مستندات، در صورتی که آمادگی ارائه محصول/سرویس/ثبت سند را دارید، گام های RequestVerification و RequestReconciliation را صورت دهید و در صورتی که آمادگی مزبور را ندارید یا کالایی برای تحویل ندارید یا موارد مشابه، با استفاده از RequestReversal وجه تراکنش را به مشتری عودت دهید.