NextPay

نکست پی

راهنمای فنی نصب ، اتصال و پیکربندی

درگاه پرداخت نکست پی

نسخه 2.0 - انتشار : 20 تیر 96

NextPay.IR

مقدمه

درگاه پرداخت نکست پی ، بستری برای انجام پرداخت های اینترنتی است. با استفاده از سرویس های متنوع نکست پی ، شما می توانید برای مشتریان خود ، دوستان خود و یا هر شخصی که لازم است به شما پول پرداخت کند ، امکان پرداخت ((اینترنتی)) را به روش های متنوع فراهم کنید .

مزایای سیستم پرداخت نکست پی :

- سوییچر هوشمند و بدون قطعی
- تسویه حساب بسیار سریع
- سازگار با انواع پلتفرم ها و زبان های برنامه نویسی
- روش های متنوع پرداخت
- امکانات پرداخت در شبکه های اجتماعی
- امکانات پرداخت آسان برای وبلاگ ها و وب سایت ها
- صفحه پرداخت شخصی ، با پروفایل پیکچر و آدرس اختصاصی برای افراد

در ادامه با تعاریف ، انواع درگاه ها ، روش اتصال به نکست پی و ارسال یک تراکنش و نمونه کد های اتصال ، آشنا می شویم .

تعاریف و واژه ها

در این راهنما ، از واژگانی استفاده شده است که بهتر است با مفهوم و منظور دقیق آن واژگان آشنا شویم :

1- **پذیرنده** : شخصیت حقیقی یا حقوقی ای که در وب سایت نکست پی ثبت نام کرده است و اقدام به دریافت خدمات پرداخت آنلاین نموده است ، از این پس ((پذیرنده)) خوانده میشود .

2- **پرداخت کننده (یا خریدار)** : کاربری که از طریق نکست پی و درگاه های بانکی متصل ، اقدام به پرداخت وجه به پذیرنده می نماید ، ((پرداخت کننده)) یا ((خریدار)) نام دارد .

3- **درگاه عامل** : عاملیت اصلی اجرای عملیات پرداخت ، هر کدام از شرکت های پرداخت الکترونیک که دارای مجوز **PSP** از بانک مرکزی جمهوری اسلامی ایران و شرکت شبکه الکترونیک پرداخت کارت "شاپرک" هستند و طرف قرارداد با نکست پی هستند و عملیات پردازش تراکنش ها تحت شبکه شتاب و بانک ها را انجام میدهند ، از این پس "درگاه عامل" خوانده میشوند . به طور مثال شرکت های پرداخت الکترونیک سامان کیش ، به پرداخت ملت ، سداد ، ایران کیش ، مبناکارت آریا ، پرداخت الکترونیک پارسیان ، پاسارگاد و ... همگی ((درگاه های عامل)) خوانده میشوند . لازم به ذکر است که علاوه بر این ها ، سایر شبکه های بین المللی ای که ممکن است در نکست پی فعال شود ، مانند وبمانی ، اسکریل ، پی پال ، ویزا ، مستر ، بیت کوین ، پرفکت مانی و ... نیز به عنوان ((درگاه عامل)) شناخته میشوند .

4- **شاپرک** : نام اختصاصی شرکت شبکه الکترونیک پرداخت کارت است که به عنوان مجری و رگولاتوری اجرای عملیات پردازش تراکنش های مبتنی بر ((کارت های شتاب)) در بانک مرکزی جمهوری اسلامی ایران تعیین شده است .

5- **نکست پی** : تمامی بستر ها ، تجهیزات ، امکانات و خدماتی که در وب سایت های **NextPay.IR** و **NextPay.ORG** ارائه و فراهم شده است ، مجموعا به طور کلی در این سند ، با نام ((نکست پی)) شناخته میشود .

6- **کلید مجوز دهی** : برای هر کدام از درگاه هایی که پذیرنده از نوع ((مستقیم)) و ((غیرمستقیم)) در نکست پی ایجاد میکند یک کلید شناسایی و اتصال ، تعریف میشود . این کلید در همه جا با عنوان **API_KEY** شناخته میشود . این کلید تعریف کننده ی این است که تراکنشی که قرار است انجام شود ، مربوط به کدام درگاه و کدام پذیرنده است و بدون این کلید امکان ایجاد یا اعتبار سنجی تراکنش ممکن نیست .

انواع درگاه ها و سرویس های پرداختی نکست پی

انواع سرویس های پرداخت در نکست پی به 2 دسته کلی تقسیم میشوند :

1- **درگاه های بانکی شتاب**

2- **سرویس های ویژه**

درگاه های بانکی شتاب ، مختص وب سایت هایی است که از زبان های برنامه نویسی مانند PHP و ASP و Python و ... استفاده می کنند و امکانات مورد نیاز برای نصب پلاگین ها ، ماژول ها و اسکریپت های پرداخت را روی هاست خود دارند . در این راهنما از ابتدای **صفحه 5** به بعد ، به طور مفصل و کامل روش های اتصال ، نمونه کد و آموزش های لازم برای استفاده از درگاه های بانکی شتاب ، توضیح داده شده است .

سرویس های ویژه ، شامل سرویس هایی است که معمولا بدون نیاز به هیچگونه امکانات و تخصص فنی ، برای انواع وب سایت ها ، وبلاگ ها ، اشخاص ، صفحات و شبکه های اجتماعی قابل استفاده است . در این راهنما ، صرفا به طور مختصر و خلاصه روش استفاده از این سرویس ها آمده است :

- **صفحه پرداخت شخصی** : در این روش پذیرنده میتواند به سادگی ، یک صفحه با آدرس اختصاصی مانند MyName.nx.plus و پروفایل پیکچر دلخواه ، برای خود ایجاد نماید . کافیست پذیرنده همین آدرس اختصاصی و کوتاه را در اختیار دیگران قرار دهد تا آنها بتوانند مبالغ دلخواه را به پذیرنده بپردازند . برای ایجاد این نوع درگاه و تعریف یک آدرس اختصاصی ، پذیرنده باید در داشبورد و پنل کاربری خود به قسمت ((امکانات ویژه)) و سپس به بخش ((صفحه پرداخت شخصی)) مراجعه نماید و با انتخاب نام دلخواه و تصویر دلخواه ، صفحه اختصاصی خود را ایجاد کند .
- **لینک پرداخت** : در این روش پذیرنده می تواند یک لینک با توضیحات و مبلغ دلخواه برای هر کدام از محصولات خود ایجاد کند و این لینک را در اختیار مشتریان و پرداخت کنندگان خود بگذارد و یا در شبکه های اجتماعی زیر محصولات مربوطه به اشتراک بگذارد . پذیرنده برای ایجاد لینک پرداخت آسان کافیست ، از پنل کاربری (داشبورد) به بخش ((امکانات ویژه)) و سپس به قسمت ((مدیریت لینک پرداخت)) مراجعه نماید و روی ((ایجاد درگاه لینک جدید)) کلیک کند و با تکمیل اطلاعات مورد نیاز ، فرم را ثبت نموده و لینک مخصوص را به طور آنی تحویل بگیرد .
- **فرم و دکمه HTML پرداخت** : پذیرنده هایی که معمولا دارای وبلاگ یا صفحات HTML هستند ، میتوانند در نکست پی یک فرم یا یک دکمه ی زیبا و ساده را با چند کلیک تولید کرده و کد HTML مربوط به آن را در جای دلخواهی از وبلاگ یا صفحه ی دلخواه وب سایت خود قرار دهند . برای ایجاد فرم یا دکمه و دریافت کد HTML

مربوط به آن کافیست پذیرنده با ورود به پنل کاربری (داشبورد) خود به بخش ((امکانات ویژه)) و سپس به قسمت ((مدیریت فرم/دکمه پرداخت)) مراجعه کند . سپس با کلیک روی ((ایجاد فرم/دکمه جدید)) ، اطلاعات مورد نیاز مانند مبلغ ، تعداد ، اطلاعات دریافتی از پرداخت کننده و ... را تنظیم کرده و درنهایت با ثبت تنظیمات ، کد HTML مربوطه را دریافت خواهید کرد که میتوانید آن را در جایگاه دلخواهی در وبلاگ یا صفحه ی HTML خود قرار دهید .

درگاه های بانکی شتاب

- **مستقیم** : در درگاه مستقیم ، پرداخت کننده مستقیماً از وب سایت پذیرنده به وب سایت بانک عامل جهت انجام تراکنش هدایت میشود .
- **غیر مستقیم** : در درگاه غیر مستقیم ، پرداخت کننده ، از وب سایت پذیرنده به صفحه ی میانی در نکست پی منتقل میشود و در این صفحه میتواند روش پرداخت را برگزیند . برخی از روش های پرداخت عبارتند از پرداخت با استفاده از کیف پول ، پرداخت با استفاده از ارزهای دیجیتال ، پرداخت با استفاده از کد های دستوری تلفن همراه و USSD (به زودی) ، پرداخت با استفاده کارت های شتاب (که در اینجا پرداخت کننده ، خود میتواند بانک عامل را انتخاب کند) . پس از اینکه در صفحه ی میانی ، پرداخت کننده روش دلخواه خود را برای پرداخت انتخاب کند ، بر اساس انتخاب وی ، مراحل تراکنش انجام میشود .

توجه : دو روش فوق ، تفاوتی از نظر کد نویسی در سایت پذیرنده ندارند . بنابراین توضیحاتی که در ادامه می آید ، برای هر 2 روش صدق میکند . پذیرنده میتواند در پنل کاربری (داشبورد) خود در نکست پی ، در بخش درگاه های بانکی شتاب ، تعیین کند که درگاه مورد نظر ((مستقیم)) باشد یا ((غیر مستقیم)) .

پیش نیاز ها

سروری که میزبانی وب سایت پذیرنده را انجام میدهد ، باید دارای یکی از 2 پکیج یا کامپوننت زیر باشد :

- **SOAP**
- **cURL** برای اجرای درخواست های GET و POST بر بستر پروتکل http

فقط یکی از 2 مورد بالا کافیست . درصورتی که از وضعیت فعال بودن یا نبودن این موارد اطلاع ندارید میتوانید با مدیر هاست یا سرور خود تماس بگیرید و موضوع را جویا شوید . این موارد بسیار رایج و عمومی هستند و معمولا روی تمامی سرورها و هاست ها به طور پیشفرض نصب و فعال هستند . همچنین راهکار جایگزین دیگری وجود دارد و آن استفاده از **NuSoap** است که در ادامهٔ این راهنما توضیح داده شده است .

سیکل کلی و چرخهٔ انجام یک تراکنش

1- درخواست ایجاد تراکنش توسط سایت پذیرنده و اخذ کد تراکنش از سرور های نکست پی

2- هدایت پرداخت کننده به همراه کد تراکنش به نکست پی و سپس بانک عامل

3- تکمیل پرداخت توسط پرداخت کننده و بازگشت از بانک عامل و هدایت وی به سایت پذیرنده

4- استعلام گیری وضعیت و صحت تراکنش از سرور های نکست پی ، توسط سایت پذیرنده . درنهایت تحویل یا عدم تحویل محصول به خریدار بر اساس وضعیت تراکنش

مرحله 1 : ایجاد تراکنش

این مرحله به منزله شروع اتصال و آغاز عملیات پرداخت است . در این مرحله ، از سرور پذیرنده باید پارامتر های زیر به سرور نکست پی ارسال شود تا برای عملیات پرداخت ((یک کد تراکنش)) صادر شود .

نام پارامتر	مثال برای مقدار	توضیحات مقدار پارامتر
api_key	d13cfbb-3e9a-4a69-8b8d-dc6574a24fb64	همان ((کلید API)) است که از طرف نکست پی برای وب سایت پذیرنده صادر شد است
order_id	123456	((شماره سفارش)) مورد نظر پذیرنده است
amount	6000	مبلغ تراکنش بر حسب ((تومان)) که باید عددی ((صحیح)) و بزرگتر از 100 تومان باشد
callback_uri	http://example.com/back.php	آدرس بازگشتی پذیرنده . پس از پایان تراکنش ، پرداخت کننده به این آدرس هدایت میشود

تمامی پارامتر های فوق و مقادیر مربوطه باید در یک آرایه با مشخص کردن ((نام پارامتر)) و ((مقدار پارامتر)) تعیین شود و سپس این آرایه در تابع TokenGenerator از آدرس http://api.nextpay.org/gateway/token.wsdl فراخوانی شود .

نکته : علاوه بر آدرس http://api.nextpay.org/gateway/token.wsdl درصورتی که از پروتکل http وصرفا از متد های POST و GET استفاده می کنید می توانید از آدرس جایگزین https://api.nextpay.org/gateway/token.http استفاده کنید .

خروجی به شکل Object در TokenGeneratorResult شامل 2 مورد خواهد بود . این موارد شامل ((trans_id)) و ((code)) میباشد . مقدار trans_id در واقع ((کد تراکنش)) می باشد و مقدار code بیانگر ((وضعیت تراکنش)) است که اگر مقدار code برابر با عدد ((1-)) باشد بیانگر این است که تراکنش ساخته شده و منتظر ارسال به بانک است . اگر کدی به غیر از 1- ارائه شود ، قابل ارسال به بانک نیست و بیانگر سایر وضعیت های تراکنش است که لیست این کد ها در ضمیمه آمده .

پیشنهاد می شود که پذیرنده در این مرحله جهت پیگیری های بعدی ، مقادیر order_id داخلی خود و amount و trans_id و code را در دیتابیس داخلی خود ذخیره کند تا در آینده بتوان تراکنش را پیگیری و بررسی کرد.

در ادامه یک مثال عملی را در زبان PHP آورده ایم

مثال در PHP برای دستور ایجاد تراکنش :

```
$parameters = array ( "api_key"=> $api_id,
                      "order_id"=> $order_id,
                      "amount"=> $price,
                      "callback_uri"=> $callback  );

    $client = new SoapClient('http://api.nextpay.org/gateway/token.wsdl');
    $token_obj = $client->TokenGenerator($parameters);
    $trans_id=$token_obj->TokenGeneratorResult->trans_id;
    $code=$token_obj->TokenGeneratorResult->code;
```

مرحله 2 : ارسال به بانک

اگر در مرحله قبل ، کد بازگشتی برابر با 1- باشد، پارامتر trans_id موجود در پاسخ را باید به انتهای آدرس زیر اضافه کرد و پرداخت کننده را به آن آدرس هدایت نمود :

http://api.nextpay.org/gateway/payment/**********

مثال : http://api.nextpay.org/gateway/payment/1246f3ea-0f6c-4806-b586-a88bc63ca76f

می توان با کد ساده جاوا اسکریپت ، پرداخت کننده را به آدرس فوق هدایت کرد :

مثال برای هدایت خریدار به درگاه عامل و صفحه بانک :

```
<script>window.location='http://api.nextpay.org/gateway/payment/**********';
</script>
```

پس از این کار ، پرداخت کننده به بانک عامل منتقل میشود و باید اطلاعات خود را وارد کرده و خرید را انجام دهد .

مرحله 3 : تکمیل پرداخت توسط پرداخت کننده

در این مرحله ، خریدار در سایت بانک و درگاه عامل مربوطه ، با وارد کردن اطلاعات کارت خود ، خرید را انجام داده و پرداخت را تکمیل می کند . جهت رعایت موارد امنیتی ، همیشه به پرداخت کنندگان یادآوری کنید که در این مرحله در آدرس بار مرورگر خود باید آدرسی دقیقا برابر با ***.shaparak.ir/*** مشاهده کنند و به نحوه نوشته شدن کلمه شاپرک دقت کنند تا در دام کلاهبرداران و صفحات جعلی و فیشینگ گرفتار نشوند . هر آدرس دیگری غیر از shaparak.ir فاقد اعتبار و اصالت است .

پس از تکمیل پرداخت ، نکست پی ، خریدار را به سایت پذیرنده هدایت میکند .

مرحله 4 : انتقال خریدار به سایت پذیرنده ، استعلام وضعیت و بررسی صحت پرداخت

در این بخش از عملیات، پس از موفق یا ناموفق بودن خرید ، پرداخت کننده به سایت پذیرنده (callback_uri) به همراه 2 پارامتر trans_id و order_id از طریق متد POST فرستاده میشود. در این مرحله پیشنهاد میشود که پذیرنده ، وجود یا عدم وجود 2 پارامتر مذکور را در دیتابیس خود بررسی کند (زیرا این 2 پارامتر همان 2 پارامتری هستند که در مرحله 1 پیشنهاد شد که آن ها را در دیتابیس خود پیش از ارسال به بانک ، وارد کنید تا بتوان در آینده تراکنش را پیگیری و بررسی کرد . بنابراین اگر در مرحله 1 آنها را ذخیره کرده باشید ، در این مرحله (چهارم) می توانید بررسی کنید که آیا قبلا این مقادیر در دیتابیس وجود دارند یا نه ؟ که اگر وجود نداشته باشند به این معنی است که این تراکنش را شما ایجاد نکرده اید و تراکنش توسط کاربر دستکاری شده و معیوب است و میتوانید همینجا فرآیند خرید را متوقف کرده و آن را ناموفق اعلام کنید !!! و اگر آن 2 پارامتر وجود داشته باشد در واقع شما مطمئن میشوید که این تراکنش قطعا مربوط به شماست و می توانید فرایند را به درستی ادامه دهید)

در ادامه باید بلافاصله صحت تراکنش و وضعیت موفق بودن یا نبودن تراکنش را از سرور های نکست پی استعلام بگیرید .برای استعلام گرفتن از وضعیت پرداخت و بررسی موفق بودن یا نبودن آن ، باید پارمتر های زیر را به نکست پی ارسال کنید تا نکست پی به عنوان پاسخ ، وضعیت پرداخت مورد نظر را به شما ارسال کند :

نام پارامتر	مثال برای مقدار	توضیحات مقدار پارامتر
api_key	d13cfbb-3e9a-4a69-8b8d-dc6574a24fb64	همان ((کلید مجوز دهی)) است که از طرف نکست پی برای درگاه پذیرنده صادر شد است
order_id	123456	((شماره سفارش)) مورد نظر پذیرنده
amount	6000	مبلغ تراکنش به ((تومان)) که باید ((صحیح)) باشد
trans_id	f3ea-0f6c-4806-b586-a88bc63ca76f1246	کد تراکنش مورد نظر

توجه :

تمامی پارامتر های فوق و مقادیر مربوطه باید در یک آرایه با مشخص کردن ((نام پارامتر)) و ((مقدار پارامتر)) تعیین شود و سپس این آرایه در تابع PaymentVerification از آدرس http://api.nextpay.org/gateway/verify.wsdl فراخوانی شود .

نکته : علاوه بر آدرس http://api.nextpay.org/gateway/verify.wsdl درصورتی که از پروتکل http وصرفا از متد های POST و GET استفاده می کنید می توانید از آدرس جایگزین http://api.nextpay.org/gateway/verify.http استفاده کنید .

خروجی به شکل Object در PaymentVerificationResult شامل پارامتری با نام ((code)) میباشد این مقدار بیانگر ((وضعیت تراکنش)) است که اگر مقدار code برابر با عدد صفر ((0)) باشد بیانگر این است که تراکنش موفقیت آمیز بوده است و مبلغ آن از حساب پرداخت کننده کسر شده و در موجودی شما (پذیرنده) افزوده شده است . اگر کدی به غیر از 0 ارائه شود ، بیانگر خطا و دلایل ناموفق بودن تراکنش است که لیست این کد ها در ضمیمه آمده است .

پیشنهاد می شود که پذیرنده در این مرحله ، وضعیت تراکنش مورد نظر را در دیتابیس خود جستجو کند و برای پرداختی که قبلا یک بار با موفقیت انجام شده و تکراری است ، اجازه ندهد که پرداخت کننده ، محصول یا خدمات مورد نظر را ((بارها و بارها)) با رفرش کردن صفحه دریافت کند .

مثال در PHP برای دستور استعلام وضعیت تراکنش :

```
$parameters = array ( "api_key"=> $api_id,
                      "order_id"=> $order_id,
                      "amount"=> $price,
                      "trans_id"=> $trans_id  );

     $client = new SoapClient('http://api.nextpay.org/gateway/verify.wsdl');
     $verify_obj = $client->PaymentVerification($parameters);
     $code =$verify_obj->PaymentVerificationResult->code;

If ($code=="0")
                {     echo "تراکنش باموفقیت انجام شد";
                } else {          echo "تراکنش ناموفق بود";          }
```

در ادامه ، ضمیمه ها و پیوست های مورد نیاز درج شده است .

پیوست اول : توضیحات تکمیلی

مثال های درج شده ، با استفاده از SOAP و در زبان PHP نوشته شده اند . درصورتی که امکان استفاده از SOAP را ندارید ، میتوانید با کمک cURL و متد های POST و GET از پروتکل http استفاده کنید . در استفاده از روش cURL آدرس مورد نیاز برای مرحله 1 (ایجاد تراکنش) https://api.nextpay.org/gateway/token.http و برای مرحله 4 (استعلام وضعیت تراکنش) http://api.nextpay.org/gateway/verify.http است .

همچنین راهکار جایگزین دیگر ، استفاده از روش NuSOAP است که در این روش نیاز به فعال بودن هیچگونه اکستنشن خاصی در PHP نیست . توسعه دهندگان و برنامه نویسان میتوانند با مراجعه به وبگاه پروژه مذکور ، آخرین نسخه آنرا دانلود و استفاده نمایند : https://sourceforge.net/projects/nusoap

نکات مهم و پیشنهادی در ذخیره سازی ، پردازش و استعلام گیری وضعیت :

1- پیش از شروع تراکنش پیشنهاد می شود در دیتابیس خود یا در سیستم خود، برای هر سفارش ، حداقل محلی یا فیلد هایی برای ذخیره سازی ((نام یا شناسه ای برای خریدار / شماره سفارش / مبلغ سفارش / کد تراکنش / زمان ایجاد فاکتور / زمان ایجاد تراکنش / وضعیت تراکنش)) در نظر بگیرید تا در هر زمانی بتوان براحتی تراکنش و سفارش مورد نظر مشتری خود را پیگیری و بررسی کنید

2- با توجه به اینکه امکان ایجاد تراکنش با مبلغ کمتر از 100 تومان وجود ندارد ، از ایجاد سفارشات و تراکنش های با مبلغ کمتر از 100 تومان خوددار ی کنید و حتما مبلغ را پیش از ارسال پارامتر ها چک کنید و مطمئن شوید که بزرگتر از 100 تومان است

3- در آغاز مرحله ی اول (مرحله درخواست ایجاد تراکنش) ، پس از ارسال پارامتر های لازم به نکست پی ، پاسخی را که از نکست پی دریافت می کنید (کد تراکنش و کد وضعیت تراکنش) در دیتابیس خود ذخیره کنید و پیش از هدایت کاربر به بانک ، حتما چک کنید که کد وضعیت تراکنش ((1-)) باشد

4- در مرحله بازگشت از بانک ((آخرین مرحله پرداخت)) حتما order_id دریافتی از نکست پی را ابتدا در دیتابیس خود جستجو کنید که مطمئن شوید در دیتابیس شما موجود است و جعلی نباشد . زیرا برخی از افراد خرابکار یا بدافزار ها یا هکر ها میتوانند کد های دیگری برای شما ارسال کنند بنابراین شما باید مطمئن شوید که این order_id در دیتابیس شما موجود است .

5- اگر به نکته ی شماره 1 عمل کرده اید ، در مرحله ی آخر order_id را در دیتابیس خود جستجو کنید و کد تراکنش مربوط به آن را ((از دیتابیس خود)) بخوانید و با کد تراکنش ارسالی از نکست پی (trans_id) مطابقت دهید و مطمئن شوید که یکسان است .

6- حتما در مرحله آخر چک کنید که order_id یا trans_id که از نکست پی دریافت کرده اید ، در دیتابیس شما از قبل دارای یک وضعیت پایان یافته نباشد . منظور این است که اجازه ندهید پرداخت کننده ، فقط با یک تراکنش ، بارها و بارها از شما محصول مورد نظر را دریافت کند ، یعنی مطمئن شوید که اگر سفارش مورد نظر قبلا یکبار به مشتری شما تحویل شده یا انجام شده یا ... ، بار دیگر به مشتری شما تحویل نشود تا دچار زیان و خسارت نشوید . به طور خلاصه چک کنید که پرداخت مورد نظر تکراری نباشد یا قبلا انجام نشده باشد !

7- پس از بررسی نکات 5 و 6 ، کد تراکنش و شماره سفارش و مبلغ و کلید api_key را به نحوی که در مرحله پایانی تراکنش قبلا توضیح دادیم ، به نکست پی ارسال کنید و پارامتر code را از پاسخی که از نکست پی دریافت می کنید بررسی کنید . دقت فرمایید این کار یک ((استعلام)) است . پس قطعا می توانید آن را بار ها و بارها تکرار کنید . اگر پاسخ عدد 0 بود ، معنی آن اینست که تراکنش با trans_id و order_id و مبلغ ارسالی دارای تایید است و با موفقیت خاتمه یافته است . اگر پاسخ ، هر عددی غیر از 0 باشد یعنی تراکنش مورد نظر دارای وضعیت ناموفق یا درحال انجام یا ... است که توضیح کد های خطا را میتوانید در پیوست شماره 2 ببینید .

8- در برنامه نویسی و کدهای برنامه ی خود موارد خطرناکی نظیر SQL Injection را حتما مورد توجه قرار دهید تا دچار مشکلات امنیتی مربوطه نشوید

9- حتی الامکان سعی کنید با کارشناسان نکست پی ، تمامی موارد فنی را بررسی کنید تا از بروز مشکلات مختلف پیشگیری کنید. کارشناسان نکست پی به طور رایگان آماده کمک رسانی به شما هستند

پیوست دوم : کد های وضعیت و خطا

شماره کد	توضیحات
0	پرداخت موفق
-1	تراکنش در وضعیت آماده برای ارسال به بانک است
-2	تراکنش به بانک ارسال شده و درحال پرداخت توسط خریدار است
-3	هنوز پاسخی در خصوص نتیجه تراکنش از بانک دریافت نشده است
-4	تراکنش توسط پرداخت کننده کنسل شده است
-20	کلید مجوزدهی (api_key) ارسال نشده است (یا مقدار پارامتر مورد نظر خالی است)
-21	شماره تراکنش (trans_id) ارسال نشده یا خالی ارسال شده است
-22	مبلغ (amount) ارسال نشده است
-23	مسیر بازگشت (callback_uri) ارسال نشده است
-24	مقدار عددی مبلغ صحیح نیست
-25	شماره تراکنش (trans_id) دوباره ارسال شده یا قابل پرداخت نیست
-26	شماره تراکنش (trans_id) ارسال نشده است
-30	مبلغ کمتر از 100 تومان است
-32	ساختار مسیر بازگشت صحیح نیست
-33	کلید مجوزدهی (api_key) صحیح نیست
-34	شماره تراکنش (trans_id) صحیح نیست

-35	نوع کلید مجوزدهی (مانند لینک، مستقیم و ...) صحیح نیست
-36	شماره سفارش (order_id) ارسال نشده یا بیش از 32 کاراکتر است
-37	تراکنش موجود نیست
-38	شماره توکن یافت نشد
-39	کلید مجوزدهی یافت نشد
-40	کلید مجوزدهی مسدود شده است
-41	پارامتر های ارسالی از طرف بانک صحیح نیست
-42	سیستم پرداخت در نکست پی دچار مشکل شده است
-43	درگاه پرداختی برای انجام روال بانکی یافت نشده است
-44	بانک عامل پاسخگو نبوده است
-45	سیستم پرداخت در نکست پی غیر فعال شده است
-46	درخواست ارسالی اشتباه است یا در نکست پی تعریف نشده است
-48	نرخ کمیسیون تعیین نشده است
-49	تراکنش یکبار انجام شده و دوباره قابل انجام نیست
-50	حساب کاربری یافت نشد
-51	کاربری در سیستم یافت نشد

پیوست سوم : نمونه کد بصورت کلاس

مثال در PHP :

```php
<?php
/**
 * Created by NextPay.ir
 * author: Nextpay Company
 * ID: @nextpay
 * Date: 09/22/2016
 * Time: 5:05 PM
 * Website: NextPay.ir
 * Email: info@nextpay.ir
 * @copyright 2016
 * @package NextPay_Gateway
 * @version 1.0
 */
class Nextpay_Payment
{
    //----- payment properties
    public $api_key = "xxxx-xxxx-xxxx-xxxx";
    public $order_id = "123456789";
    public $amount = 100;
    public $trans_id = "";
    public $params = array();
    public $server_soap = "https://api.nextpay.org/gateway/token.wsdl";
    public $server_http = "https://api.nextpay.org/gateway/token.http";
    public $request_http = "https://api.nextpay.org/gateway/payment";
    public $request_verify_soap =
"https://api.nextpay.org/gateway/verify.wsdl";
    public $request_verify_http =
"https://api.nextpay.org/gateway/verify.http";
    public $callback_uri = "http://example.com";
    private $keys_for_verify =
array("api_key","order_id","amount","callback_uri");
    private $keys_for_check = array("api_key","order_id","amount","trans_id");
    //----- controller properties
    public $default_verify = Type_Verify::SoapClient;
    /**
     * Nextpay_Payment constructor.
     * @param array|bool $params
     * @param string|bool $api_key
     * @param string|bool $order_id
```

```
     * @param string|bool $url
     * @param int|bool $amount
     */
    public function __construct($params=false, $api_key=false, $order_id=false,
$amount=false, $url=false)
    {
        $trust = true;
        if(is_array($params))
        {
            foreach ($this->keys_for_verify as $key )
            {
                if(!array_key_exists($key,$params))
                {
                    $error = "<h2>آرایه ارسالی دارای مشکل میباشد.</h2>";
                    $error .= "<h4>نمونه مثال برای آرایه ارسالی.</h4>";
                    $error .= /** @lang text */
                        "<pre>
                        array(\"api_key\"=>\"شناسه api\",
                              \"order_id\"=>\"شماره فاکتور\",
                              \"amount\"=>\"مبلغ\",
                              \"callback_uri\"=>\"مسیر باگشت\")
                        </pre>";
                    $trust = false;
                    $this->show_error($error);
                    break;
                }
            }
            if($trust)
            {
                $this->params = $params;
                $this->api_key = $params['api_key'];
                $this->order_id = $params['order_id'];
                $this->amount = $params['amount'];
                $this->callback_uri = $params['callback_uri'];
            }
            else
            {
                $this->show_error("برای مقدارهی پارامتر ها باید بصورت آرایه اقدام نمایید");
                exit("End with Error!!!");
            }
```

```
        }
        else
        {
            if($api_key)
                $this->api_key = $api_key;
            //else
            //    $this->show_error("مقدار دهی نشده است  api  شناسه مربوط به");
            if($order_id)
                $this->order_id = $order_id;
            //else
            //    $this->show_error("شماره فاکتور مقداردهی نشده است");
            if($amount)
                $this->amount = $amount;
            //else
            //    $this->show_error("مبلغ تعیین نشده است");
            if($url)
                $this->callback_uri = $url;
            //else
            //    $this->show_error("مسیر بازگشت تعیین نشده است");
            $this->params = array(
                "api_key"=>$this->api_key,
                "order_id"=>$this->order_id,
                "amount"=>$this->amount,
                "callback_uri"=>$this->callback_uri);
        }
    }
    /**
     * @return string
     * return trans_id
     */
    public function token()
    {
        $res = "";
        switch ($this->default_verify)
        {
            case Type_Verify::SoapClient:
                try
                {
                    $soap_client = new SoapClient($this->server_soap,
array('encoding' => 'UTF-8'));
                    $res = $soap_client->TokenGenerator($this->params);
```

```
                    $res = $res->TokenGeneratorResult;
                    if ($res != "" && $res != NULL && is_object($res)) {
                        if (intval($res->code) == -1)
                            $this->trans_id = $res->trans_id;
                        /*else
                            $this->code_error($res->code);*/
                    }
                    else
                        $this->show_error("خطا در پاسخ دهی به درخواست با
SoapClinet");
                }
                catch(Exception $e){
                    $this->show_error($e->getMessage());
                }
                break;
            case Type_Verify::NuSoap:
                try
                {
                    include_once ("include/nusoap/nusoap.php");
                    $client = new nusoap_client($this->server_soap,'wsdl');
                    $error = $client->getError();
                    if ($error)
                        $this->show_error($error);
                    $res = $client->call('TokenGenerator',array($this-
>params));
                    if ($client->fault)
                    {
                        echo "<h2>Fault</h2><pre>";
                        print_r ($res);
                        echo "</pre>";
                        exit(0);
                    }
                    else
                    {
                        $error = $client->getError();
                        if ($error)
                            $this->show_error($error);
                        $res = $res['TokenGeneratorResult'];
                        if ($res != "" && $res != NULL && is_array($res)) {
                            if (intval($res['code']) == -1) {
                                $this->trans_id = $res['trans_id'];
                                $res = (object)$res;
                            }/*else
```

```
                                $this->code_error($res['code']);*/
                        }
                        else
                            $this->show_error("خطا در پاسخ دهی به درخواست با
NuSoap_Client");
                    }
                }
                catch(Exception $e){
                    $this->show_error($e->getMessage());
                }
                break;
            case Type_Verify::Http:
                try
                {
                    if( !$this->cURLcheckBasicFunctions() ) $this-
>show_error("UNAVAILABLE: cURL Basic Functions");
                    $curl = curl_init();
                    curl_setopt($curl, CURLOPT_URL, $this->server_http);
                    curl_setopt($curl, CURLOPT_POST, true);
                    curl_setopt($curl, CURLOPT_FOLLOWLOCATION, true);
                    curl_setopt($curl, CURLOPT_SSL_VERIFYHOST, false);
                    curl_setopt($curl, CURLOPT_SSL_VERIFYPEER, false);
                    curl_setopt($curl, CURLOPT_POSTFIELDS,
                        "api_key=".$this->api_key."&order_id=".$this-
>order_id."&amount=".$this->amount."&callback_uri=".$this->callback_uri);
                    curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
                    /** @var int | string $server_output */
                    $res = json_decode(curl_exec ($curl));
                    curl_close ($curl);
                    if ($res != "" && $res != NULL && is_object($res)) {
                        if (intval($res->code) == -1)
                            $this->trans_id = $res->trans_id;
                        /*else
                            $this->code_error($res->code);*/
                    }
                    /*else
                        $this->show_error("خطا در پاسخ دهی به درخواست با Curl");*/
                }
                catch (Exception $e){
                    $this->show_error($e->getMessage());
                }
                break;
```

```
            default:
                try
                {
                    $soap_client = new SoapClient($this->server_soap,
array('encoding' => 'UTF-8'));
                    $res = $soap_client->TokenGenerator($this->params);
                    $res = $res->TokenGeneratorResult;
                    if ($res != "" && $res != NULL && is_object($res)) {
                        if (intval($res->code) == -1)
                            $this->trans_id = $res->trans_id;
                        /*else
                            $this->code_error($res->code);*/
                    }
                    else
                        $this->show_error("خطا در پاسخ دهی به درخواست با
SoapClinet");
                }
                catch(Exception $e){
                    $this->show_error($e->getMessage());
                }
                break;
        }
        return $res;
    }
    /**
     * @param string $trans_id
     */
    public function send($trans_id)
    {
        if(isset($trans_id))
        {
            header_remove();
            ob_clean();
            if (headers_sent()) {
                echo "<script> location.replace(\"".$this-
>request_http."/$trans_id"."\"); </script>";
            }
            else
            {
                header('Location: '.$this->request_http."/$trans_id");
                exit(0);
            }
        }
```

```
        else
            $this->show_error("empty trans_id param send");
    }
    /**
     * @param array|bool $params
     * @param string|bool $api_key
     * @param string|bool $trans_id
     * @param int|bool $amount
     * @param string|bool $order_id
     * @return int|mixed
     */
    public function verify_request($params=false, $api_key=false,
$order_id=false, $trans_id=false, $amount=false)
    {
        $res = 0;
        $trust = true;
        if(is_array($params))
        {
            foreach ($this->keys_for_check as $key )
            {
                if(!array_key_exists($key,$params))
                {
                    $error = "<h2>آرایه ارسالی دارای مشکل میباشد.</h2>";
                    $error .= "<h4>نمونه مثال برای آرایه ارسالی.</h4>";
                    $error .= /** @lang text */
                        "<pre>
                            array(\"api_key\"=>\"شناسه api\",
                                  \"order_id\"=>\"شماره فاکتور\",
                                  \"amount\"=>\"مبلغ\",
                                  \"trans_id\"=>\"شماره تراکنش\")
                        </pre>";
                    $trust = false;
                    $this->show_error($error);
                    break;
                }
            }
            if($trust)
            {
                $this->trans_id = $params['trans_id'];
                $this->api_key = $params['api_key'];
```

```
            $this->order_id = $params['order_id'];
            $this->amount = $params['amount'];
        }
        else
        {
            $this->show_error("برای مقدارهی پارامتر ها باید بصورت آرایه اقدام نمایید");
            exit("End with Error!!!");
        }
    }
    if($api_key){
        $this->api_key = $api_key;
        $this->params['api_key'] = $api_key;
    }elseif (isset($this->api_key)) {
        $this->params['api_key'] = $this->api_key;
    }
    //else
    //    $this->show_error("مقدار دهی نشده است  api شناسه مربوط به");
    if($order_id){
        $this->order_id = $order_id;
        $this->params['order_id'] = $order_id;
    }elseif (isset($this->order_id)){
        $this->params['order_id'] = $this->order_id;
    }
    //else
    //    $this->show_error("شماره فاکتور مقداردهی نشده است");
    if($amount){
        $this->amount = $amount;
        $this->params['amount'] = $amount;
    }elseif (isset($this->amount)){
        $this->params['amount'] = $this->amount;
    }
    //else
    //    $this->show_error("مبلغ تعیین نشده است");
    if($trans_id){
        $this->trans_id = $trans_id;
        $this->params['trans_id'] = $trans_id;
    }elseif (isset($this->trans_id)){
        $this->params['trans_id'] = $this->trans_id;
    }
    //else
    //    $this->show_error("شماره نراکنش تعیین نشده است");
```

```
        switch ($this->default_verify)
        {
            case Type_Verify::SoapClient:
                try
                {
                    $soap_client = new SoapClient($this->request_verify_soap,
array('encoding' => 'UTF-8'));
                    $res = $soap_client->PaymentVerification($this->params);
                    $res = $res->PaymentVerificationResult;
                    if ($res != "" && $res != NULL && is_object($res)) {
                        $res = $res->code;
                    }
                    else
                        $this->show_error("خطا در پاسخ دهی به درخواست با
SoapClinet");
                }
                catch(Exception $e){
                    $this->show_error($e->getMessage());
                }
                break;
            case Type_Verify::NuSoap:
                try
                {
                    include_once ("include/nusoap/nusoap.php");
                    $client = new nusoap_client($this->server_soap,'wsdl');
                    $error = $client->getError();
                    if ($error)
                        $this->show_error($error);
                    $res = $client->call('PaymentVerification',array($this-
>params));
                    if ($client->fault)
                    {
                        echo "<h2>Fault</h2><pre>";
                        print_r ($res);
                        echo "</pre>";
                        exit(0);
                    }
                    else
                    {
                        $error = $client->getError();
                        if ($error)
                            $this->show_error($error);
                        $res = $res['PaymentVerificationResult'];
```

```
                    if ($res != "" && $res != NULL && is_array($res)) {
                        $res = $res['code'];
                    }
                    else
                        $this->show_error("خطا در پاسخ دهی به درخواست با
NuSoap_Client");
                }
            }
            catch(Exception $e){
                $this->show_error($e->getMessage());
            }
            break;
        case Type_Verify::Http:
            try
            {
                if( !$this->cURLcheckBasicFunctions() ) $this-
>show_error("UNAVAILABLE: cURL Basic Functions");
                $curl = curl_init();
                curl_setopt($curl, CURLOPT_URL, $this-
>request_verify_http);
                curl_setopt($curl, CURLOPT_POST, 1);
                curl_setopt($curl, CURLOPT_FOLLOWLOCATION, true);
                curl_setopt($curl, CURLOPT_SSL_VERIFYHOST, false);
                curl_setopt($curl, CURLOPT_SSL_VERIFYPEER, false);
                curl_setopt($curl, CURLOPT_POSTFIELDS,
                    "api_key=".$this->api_key."&order_id=".$this-
>order_id."&amount=".$this->amount."&trans_id=".$this->trans_id);
                curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
                /** @var int | string $server_output */
                $res = json_decode(curl_exec ($curl));
                curl_close ($curl);
                if ($res != "" && $res != NULL && is_object($res)) {
                    $res = $res->code;
                }
                else
                    $this->show_error("خطا در پاسخ دهی به درخواست با Curl");
            }
            catch (Exception $e){
                $this->show_error($e->getMessage());
            }
            break;
        default:
```

```
                try
                {
                    $soap_client = new SoapClient($this->request_verify_soap,
array('encoding' => 'UTF-8'));
                    $res = $soap_client->PaymentVerification($this->params);
                    $res = $res->PaymentVerificationResult;
                    if ($res != "" && $res != NULL && is_object($res)) {
                        $res = $res->code;
                    }
                    else
                        $this->show_error("خطا در پاسخ دهی به درخواست با
SoapClinet");
                }
                catch(Exception $e){
                    $this->show_error($e->getMessage());
                }
                break;
        }
        return $res;
    }
    /**
     * @param string | string $error
     */
    public function show_error($error)
    {
        echo "<h1>وقوع خطا !!!</h1>";
        echo "<h4>{$error}</h4>";
    }
    /**
     * @param int | string $error_code
     */
    public function code_error($error_code)
    {
        $error_code = intval($error_code);
        $error_array = array(
            0 => "Complete Transaction",
          -1 => "Default State",
          -2 => "Bank Failed or Canceled",
          -3 => "Bank Payment Pendding",
          -4 => "Bank Canceled",
          -20 => "api key is not send",
          -21 => "empty trans_id param send",
```

```
          -22 => "amount in not send",
          -23 => "callback in not send",
          -24 => "amount incorrect",
          -25 => "trans_id resend and not allow to payment",
          -26 => "Token not send",
          -30 => "amount less of limite payment",
          -32 => "callback error",
          -33 => "api_key incorrect",
          -34 => "trans_id incorrect",
          -35 => "type of api_key incorrect",
          -36 => "order_id not send",
          -37 => "transaction not found",
          -38 => "token not found",
          -39 => "api_key not found",
          -40 => "api_key is blocked",
          -41 => "params from bank invalid",
          -42 => "payment system problem",
          -43 => "gateway not found",
          -44 => "response bank invalid",
          -45 => "payment system deactived",
          -46 => "request incorrect",
          -48 => "commission rate not detect",
          -49 => "trans repeated",
          -50 => "account not found",
          -51 => "user not found"
        );

        if (array_key_exists($error_code, $error_array)) {
                                                    return $error_array[$error_code];
        } else {
                                                   return "error code : $error_code";
      }
    }
    /**
     * @return bool
     */
    public function cURLcheckBasicFunctions()
    {
        if( !function_exists("curl_init") &&
            !function_exists("curl_setopt") &&
            !function_exists("curl_exec") &&
            !function_exists("curl_close") ) return false;
        else return true;
```

```
    }
    /**
     * @return int
     */
    public function getAmount()
    {
        return $this->amount;
    }
    /**
     * @return string
     */
    public function getApiKey()
    {
        return $this->api_key;
    }
    /**
     * @return string
     */
    public function getOrderId()
    {
        return $this->order_id;
    }
    /**
     * @return string
     */
    public function getCallbackUri()
    {
        return $this->callback_uri;
    }
    /**
     * @return string
     */
    public function getTransId()
    {
        return $this->trans_id;
    }
    /**
     * @return array
     */
    public function getParams()
    {
        return $this->params;
    }
```

```
    /**
     * @param int|int $amount
     */
    public function setAmount($amount)
    {
        $this->amount = $amount;
        $this->params['amount'] = $this->amount;
    }
    /**
     * @param bool|string $api_key
     */
    public function setApiKey($api_key)
    {
        $this->api_key = $api_key;
        $this->params['api_key'] = $this->api_key;
    }
    /**
     * @param bool|string $order_id
     */
    public function setOrderId($order_id)
    {
        $this->order_id = $order_id;
        $this->params['order_id'] = $this->order_id;
    }
    /**
     * @param string $trans_id
     */
    public function setTransId($trans_id)
    {
        $this->trans_id = $trans_id;
        $this->params['trans_id'] = $this->trans_id;
    }
    /**
     * @param string|string $callback_uri
     */
    public function setCallbackUri($callback_uri)
    {
        $this->callback_uri = $callback_uri;
        $this->params['callback_uri'] = $this->callback_uri;
    }
    /**
     * @param array|array $params
     */
```

```
    public function setParams($params)
    {
        $trust = true;
        if(is_array($params))
        {
            if (isset($this->keys_for_verify))
            {
                foreach ($this->keys_for_verify as $key )
                {
                    if(!array_key_exists($key,$params))
                    {
                        $trust = false;
                        $error = "<h2>آرایه ارسالی دارای مشکل میباشد.</h2>";
                        $error .= "<h4>نمونه مثال برای آرایه ارسالی.</h4>";
                        $error .= /** @lang text */
                            "<pre>
                                array(\"api_key\"=>\"شناسه api\",
                                      \"order_id\"=>\"شماره فاکتور\",
                                      \"amount\"=>\"مبلغ\",
                                      \"callback_uri\"=>\"مسیر باگشت\")

                            </pre>";
                        $this->show_error($error);
                        break;
                    }
                }
            }
            else
                $this->show_error("برای مقدارهی پارامتر ها باید بصورت آرایه اقدام نمایید");
            if($trust)
            {
                $this->params = $params;
                $this->api_key = $params['api_key'];
                $this->order_id = $params['order_id'];
                $this->amount = $params['amount'];
                $this->callback_uri = $params['callback_uri'];
            }
            else
                $this->show_error("برای مقدارهی پارامتر ها باید بصورت آرایه اقدام نمایید");
        }
```

```
        else
            $this->show_error("برای مقدارهی پارامتر ها باید بصورت آرایه اقدام نمایید");
    }
    /**
     * @param int $default_verify
     */
    public function setDefaultVerify($default_verify)
    {
        switch ($default_verify){
            case 0:
            case Type_Verify::NuSoap:
                $this->default_verify = Type_Verify::NuSoap;
                break;
            case 1:
            case Type_Verify::SoapClient:
                $this->default_verify = Type_Verify::SoapClient;
                break;
            case 2:
            case Type_Verify::Http:
                $this->default_verify = Type_Verify::Http;
                break;
            default:
                $this->default_verify = Type_Verify::SoapClient;
        }
    }
}
class Type_Verify
{
    const NuSoap = 0;
    const SoapClient = 1;
    const Http = 2;
}
?>
```