
هاب صدور گواهی

سرویس های صدور گواهینامه

راهنمای بهره برداران



واحد مدیریت هاب صدور گواهی

۱.۳	نسخه
فروردین ماه ۱۴۰۳	تاریخ
PKI-RA-API-DG	شناسه
عمومی	طبقه بندی



فهرست مطالب

۰	فهرست مطالب
۲	تاریخچه
۲	پیوست‌ها
۳	پیشگفتار
۳	امکانات سرویس
۴	تعاریف عمومی
۵	چگونه از این سرویس می‌توان استفاده کرد؟
۵	فرآیند صدور گواهی امضای دیجیتال
۷	سرویس‌های درگاه
۸	روش امضا جهت فراخوانی سرویس
۹	سرویس‌های حوزه مرکز میانی (CA Services)
۹	GetCAList
۱۰	GetCAProfileInfo
۱۰	IsCAAvailable
۱۱	سرویس‌های حوزه گواهی (Certificate Services)
۱۱	CertificateRequest
۱۳	CertificateIssue
۱۴	KeyStoreRequest
۱۷	KeyStoreIssue
۱۸	RevokeCertificate
۱۹	AuthenticationCompleted
۲۰	RecievedCertConfirmation
۲۱	سرویس‌های گزارش‌گیری (Reporting Services)
۲۱	IssuingReport
۲۲	IsRequestAuthenticated
۲۳	پیوست شماره ۱
۲۵	پیوست شماره ۲
۲۶	پیوست شماره ۳
۲۷	پیوست شماره ۴

تاریخچه

نسخه	تاریخ	تهیه کنندگان	مرور کنندگان	توضیحات
۱.۰	۱۴۰۳/۰۱/۱۵	واحد مرکز میانی	واحد کنترل کیفیت	تهیه سند
۱.۱	۱۴۰۳/۰۲/۲۲	واحد مرکز میانی	واحد کنترل کیفیت	اضافه شدن پیوست ۴ در تولید زوج کلید
۱.۲	۱۴۰۳/۰۲/۲۹	واحد مرکز میانی	واحد کنترل کیفیت	اصلاح جدول دلایل ابطال گواهی
۱.۳	۱۴۰۳/۰۳/۰۳	واحد مرکز میانی	واحد کنترل کیفیت	اضافه شدن توضیحات به Certificate Issue و Revoke Certificate

پیوست‌ها

شماره	عنوان	نسخه	توضیحات
۱	جدول کد خطا	۱.۰	در این پیوست کدهای خطا برنامه وجود دارد
۲	فرم اخذ کد مشتری	۱.۰	لیست اطلاعاتی که برای دریافت کد مشتری باید ارسال شود
۳	فلوچارت صدور گواهی	۱.۰	نمون فلوچارت صدور گواهی بکار رفته در برنامه mKeyOne
۴	تولید زوج کلید	۱.۰	نحوه تولید زوج کلید در تصدیق هویت متقاضی

پیشگفتار

استنادپذیری اسناد و عملیات الکترونیک یکی از اساسی ترین پایه های خدمات الکترونیکی بخصوصی در حوزه هایی که مسائل حقوقی در آن وجود دارد می باشد.

طبق قوانین جمهوری اسلامی ایران تنها راه اعطای وجاهت حقوقی به یک سند الکترونیکی امضای دیجیتال آن سند است. به منظور امضای دیجیتال هر شخص باید گواهی امضا دریافت کند. در این سند نحوه استفاده از سرویس های صدور گواهی شرکت پندار کوشک ایمن به منظور دریافت گواهی امضای دیجیتال ارائه شده است.

از ویژگی های اصلی این سرویس می تواند به موارد ذیل اشاره کرد:

- ۱- دسترسی به مراکز میانی مختلف جهت صدور گواهی فقط با یک پیاده سازی
- ۲- عدم وابستگی به یک مرکز میانی خاص
- ۳- پایداری بالای سرویس
- ۴- سرعت بالای سرویس
- ۵- امکان ارائه سرویس های پایه مثل شاهکار و ثبت احوال
- ۶- کسب درآمد از صدور گواهی

امکانات سرویس

در سرویس های مرکز صدور گواهی پندار قابلیت های زیر وجود دارد:

- ۱- ثبت اطلاعات هویتی متقاضی گواهی
- ۲- صدور گواهی امضای دیجیتال برای متقاضی
- ۳- ابطال گواهی امضای دیجیتال صاحب گواهی
- ۴- دریافت لیست گواهی های امضای یک فرد
- ۵- تمدید گواهی امضای دیجیتال یک فرد
- ۶- تشخیص وضعیت مرکز صدور گواهی
- ۷- بررسی وضعیت احراز هویت متقاضی گواهی
- ۸- تایید هویت یک متقاضی گواهی

تعاریف عمومی

Customercode: کد مشتری که توسط شرکت به ایشان تخصیص داده می‌شود.

LisenceNumber: شماره مجوز که توسط شرکت به مشتری تخصیص داده می‌شود.

caName: نام یا کد مرکز صدور که باید از شرکت دریافت شود.

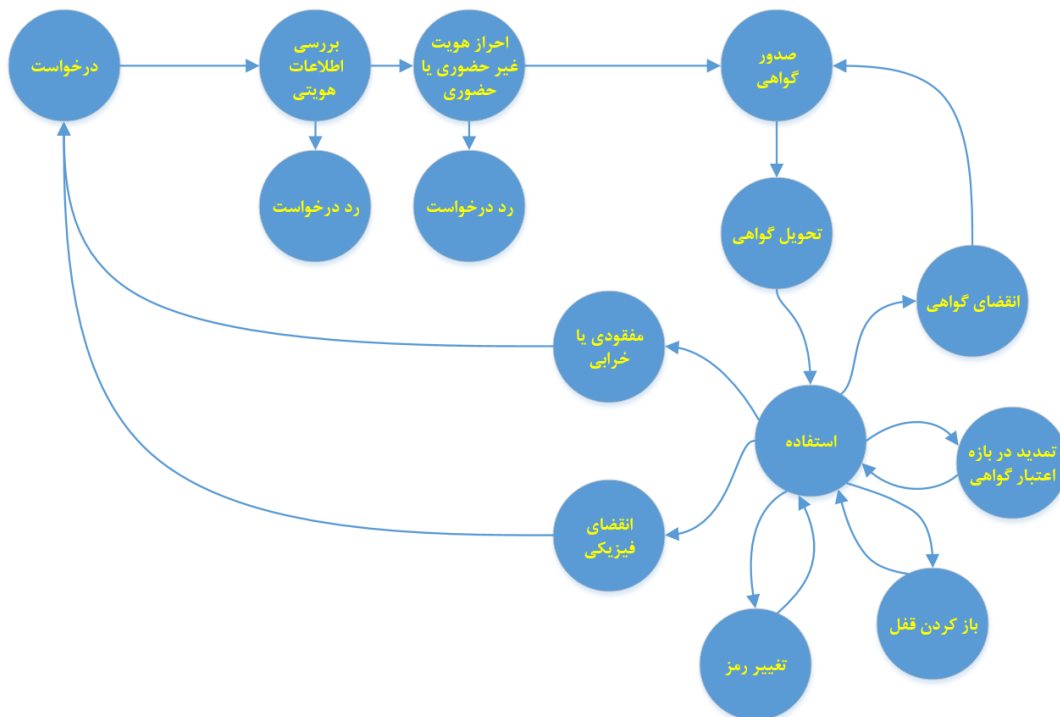
profileName: نام پروفایل گواهی که باید از شرکت دریافت شود.

متقاضی گواهی: فردی که می‌خواهد گواهی امضای دیجیتال دریافت نماید.

مشتری: شخص حقوقی طرف قرارداد شرکت پندار کوشک ایمن که از سروی‌های برای صدور گواهی به متقاضی استفاده می‌کند.

چرخه حیات گواهی:

چرخه حیات گواهی بر روی توکن یا کارت هوشمند یا برنامه موبایل Certificate Live Cycle



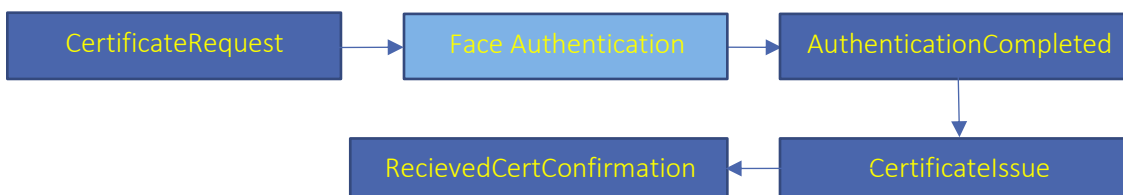
چگونه از این سرویس می توان استفاده کرد ؟

برای استفاده از سرویس‌های صدور گواهی شرکت پندار کوشک ایمن باید مراحل زیر طی شود:

انجام دهنده	اقدام	
طرفین	امضای قرارداد دفتر صدور گواهی الکترونیکی پیوست ۵	۱
طرفین	امضای توافقنامه عدم افشای اطلاعات پیوست ۶	۲
پندار کوشک ایمن	ایجاد کد مشتری و تخصیص دسترسی به سرویس مطابق با اطلاعات پیوست ۲ و همچنین تولید زوج کلید اختصاصی مطابق با فرآیند پیوست ۴	۳
مشتری	پیاده سازی سرویس در سامانه مشتری	۴
مشتری	استفاده از سرویس و کسب درآمد از آن	۵
پندار کوشک ایمن	پشتیبانی	۶

فرآیند صدور گواهی امضای دیجیتال

برای صدور گواهی امضای دیجیتال مطابق توالی زیر باید ابتدا اطلاعات متقاضی گواهی دریافت و متدهای زیر از سرویس فراخوانی شود:



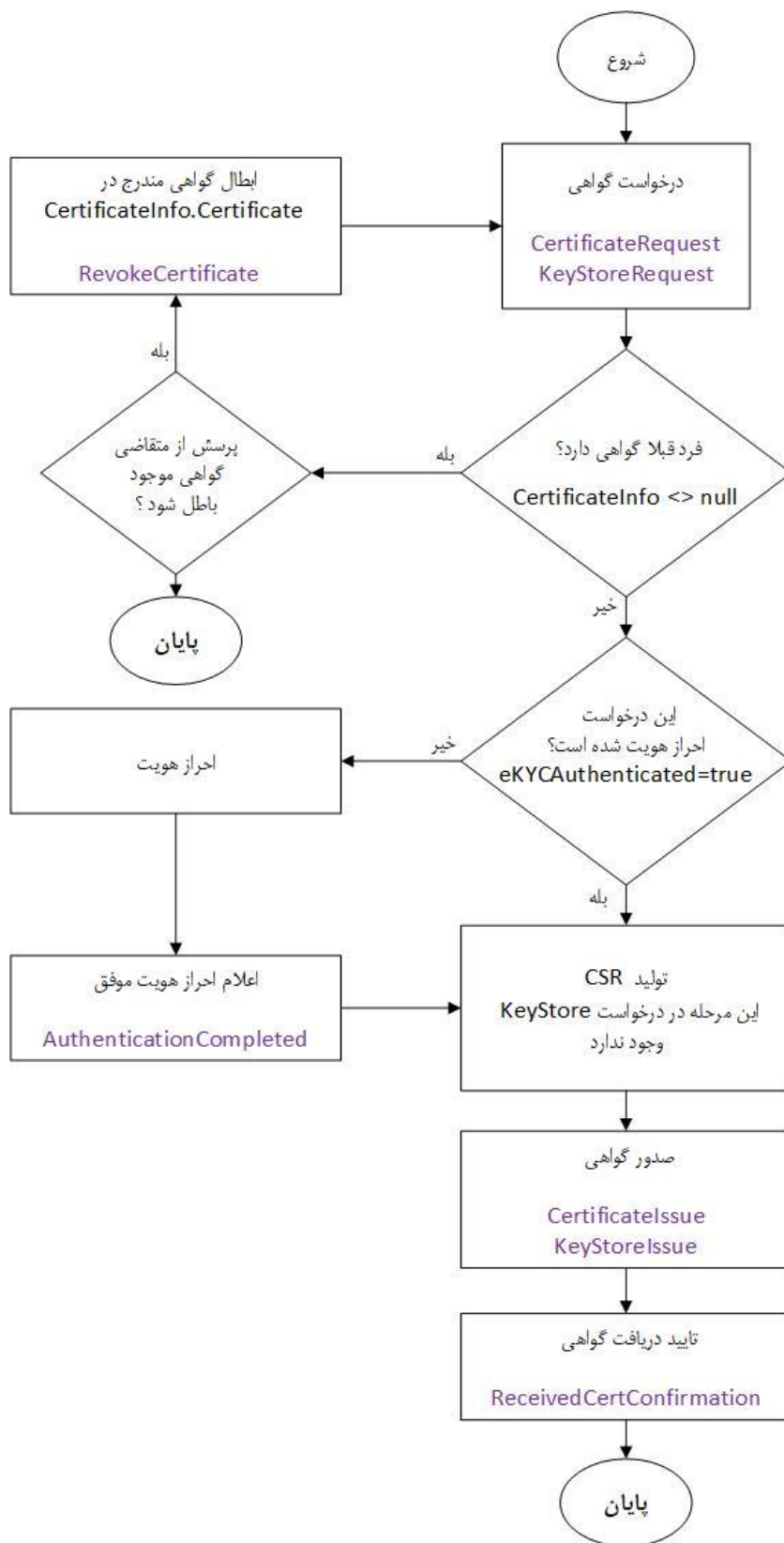
همانگونه که در نمودار فوق مشخص است، کلیات فرآیند دریافت گواهی امضای دیجیتال چنین است:

- ۱- ثبت درخواست گواهی با اطلاعات هویتی متقاضی
- ۲- احراز هویت متقاضی
- ۳- اعلام انجام موفق احراز هویت
- ۴- ارسال CSR و دریافت گواهینامه
- ۵- تایید دریافت گواهینامه

به منظور احراز هویت چهره از هر سرویس مورد تایید مرکز دولتی ریشه می تواند استفاده کرد.

فلوچارت کامل صدور گواهی

موارد بنفش نام متد در سرویس ها می باشد.





سرویس‌های درگاه

سند پیش رو مربوط به سرویس‌های درگاه RA شرکت پندار کوشک ایمن می‌باشد. جهت دریافت نگارش سرویس فعال می‌تواند از آدرس زیر استفاده کرد:
<https://api.pki.co.ir/ra/GetVersion>
کلید متدها از طریق آدرس زیر در دسترس می‌باشند:

https://api.pki.co.ir/ra/{api_name}

جهت اخذ گواهی امضای دیجیتال و مدیریت چرخه حیات گواهی متدهای زیر در سرویس این شرکت وجود دارد:

CA

- ✚ [GetCAList](#)
- ✚ [GetCAProfileInfo](#)
- ✚ [IsCAAvailable](#)

Certificate

- ✚ [CertificateRequest](#)
- ✚ [CertificateIssue](#)
- ✚ [keyStoreRequest](#)
- ✚ [KeyStoreIssue](#)
- ✚ [AuthenticationCompleted](#)
- ✚ [RevokeCertificate](#)
- ✚ [RecievedCertConfirmation](#)

Report

- ✚ [IsRequestAuthenticated](#)
- ✚ [IssuingReport](#)

نکات مهم :

- ۱- در تمام سرویس‌ها نام‌ها در JSON ورودی حساس به حروف بوده و باید دقیقاً مطابق نام درج شده در سند، JSON ورودی ساخته شود.
- ۲- دسترسی به سرویس مبتنی بر امضای اختصاصی RSA Sign هر مشتری انجام شده و فاقد Username, Password است.
- ۳- کلید متدها بر پایه تراکنش (Transaction Base) بوده و جلسه‌ای (Session) جهت انجام درخواست ایجاد نمی‌شود.
- ۴- برای هر تراکنش نیاز به تصدیق هویت متقاضی از طریق امضای اختصاصی وی (RSA Sign) در آن تراکنش است.
- ۵- هر تراکنش انجام شده با امضای مشتری بعنوان سند انکارناپذیر درخواست انجام آن تراکنش از سوی مشتری تلقی شده و سندیت حقوقی دارد و کلید مسولیت آن بعهدہ مشتری می‌باشد.

روش امضا جهت فراخوانی سرویس

به منظور احراز هویت درخواست کننده یک متد از سرویس، هر درخواست باید توسط درخواست کننده امضا شود به این ترتیب هم هویت درخواست کننده مشخص می‌شود و هم درخواست کننده نمی‌تواند منکر درخواست خود شود. برای این منظور سرویس گیرنده باید یک زوج کلید با طول ۱۰۲۴ تولید کرده (مطابق پیوست ۶) و کلید عمومی آن را در اختیار مدیریت مرکز هاب صدور گواهی قرار دهد. در زمان فراخوانی تمام سرویس‌ها لازم است بخش بدنه (body) درخواست با الگوریتم RSA و هش الگوریتم SHA1 امضا شده و نتیجه آن در متغیر Signature در سرانه (header) بسته (http) قرار گیرد. برای این منظور باید رشته بدنه (body string) که در قالب JSON می‌باشد با کدینگ UTF8 به بایت تبدیل شده و با کلید خصوصی سرویس گیرنده امضا شود. همچنین به منظور شناسایی امضا کننده بسته باید کد مشتری و لایسنس مشتری نیز در سرانه در متغیر CustomerCode درج گردد.

بعنوان مثال اگر کد مشتری (CustomerCode) برابر ۱۱۱۱۱۱۱۱۱۱ و لایسنس برابر ۱ باشد باید در سرانه مقدار زیر قرار گیرد.

```
CustomerCode= 11111111111-1  
Signature= Base64(Sign_RSAWithSHA1(UTF8.Byte(body)))
```

متدهایی که در آن‌ها امضای بسته الزامی است چنانچه بدون امضا ارسال شوند خطا خواهند داد. از آنجا که مشتری درخواست خود را بطور کامل امضا می‌کند ضمن احراز هویت و بررسی تمامیت اطلاعات ارسالی، ارسال کننده درخواست نمی‌تواند منکر درخواست خود شود. مقدار امضای Signature مطابق تابع نمونه در زیر (SignString) تولید می‌شود. یعنی محتوای JSON ورودی که در body ارسال می‌شود مطابق تابع SignString امضا شده و در متغیر Signature در سرانه (header) درخواست ارسال می‌شود. از این تابع برای امضای کلیه مقادیر رشته ای در تمام درخواست‌ها می‌تواند استفاده کرد.

نمونه کد امضای رشته به زبان C# برای امضای body یا مقادیر امضا از نوع رشته در برخی متدها

```
private string SignString(string data, X509Certificate2 cert)  
{  
    byte[] data4Sign = Encoding.UTF8.GetBytes(data)  
    if (cert.HasPrivateKey)  
    {  
        RSA rsa = cert.GetRSAPrivateKey();  
        byte[] signed= rsa.SignData(data4Sign, HashAlgorithmName.SHA1, RSASignaturePadding.Pkcs1);  
        return Convert.ToBase64String(signed);  
    }  
    return "";  
}
```

نمونه کد امضا بایت به زبان C# برای امضا مقادیر از نوع باینری در درخواست‌ها

```
private string SignBytes(byte[] data4Sign, X509Certificate2 cert)  
{  
    if (cert.HasPrivateKey)  
    {  
        RSA rsa = cert.GetRSAPrivateKey();  
        byte[] signed= rsa.SignData(data4Sign, HashAlgorithmName.SHA1, RSASignaturePadding.Pkcs1);  
        return Convert.ToBase64String(signed);  
    }  
    return null;  
}
```



سرویس های حوزه مرکز میانی (CA Services)

این سرویس دارای سه API است:

GetCAList

این متد جهت دریافت لیست مراکز میانی است که در اختیار یک مشتری می باشد.

ورودی این متد کد مشتری است و خروجی آن لیست مراکز میانی است که مشتری به آن دسترسی دارد.

GetCAList API	
Request	
Method	Get
Body Content-Type	application/json
Encoding	UTF8
Parameters	Customercode="کد مشتری"
Response	
Status Code	200 Success / 401 Unauthorized
	{ "IsSuccess": true, "CAList": [], "ErrorCode": 0, "ErrorMessage": "" }



GetCAProfileInfo

این متد لیست پروفایل‌های مجاز به استفاده یک مشتری را در یک مرکز صدور گواهی را در اختیار می‌گذارد.

ورودی این متد کد مشتری و نام مرکز میانی است که از متد GetCAList بدست آمده می‌باشد و خروجی آن لیست پروفایل‌هایی است که مشتری به آن دسترسی دارد.

GetCAProfileInfo API	
Request	
Method	Get
Body Content-Type	application/json
Encoding	UTF8
Parameters	Customercode="کد مشتری" caName="نام مرکز صدور"
Response	
Status Code	200 Success / 401 Unauthorized
	<pre>{ "IsSuccess": true, "Profile": [{ "SubjectDNConfig": "", "ProfileName": "", "Price":, "FaceAuth": true/false, "InputType": "manually/card/mobile", "Approval": false/true, "ProductId":, "Description": "" }], "ErrorCode": 0, "ErrorMessage": "" }</pre>

IsCAAvailable

این متد مشخص می‌کند یک مرکز صدور هم اکنون فعال است یا خیر

IsAvailable API	
Request	
Method	Get
Body Content-Type	application/json
Encoding	UTF8
Parameters	caName= نام مرکز میانی
Response	
Status Code	200 Success / 401 Unauthorized
	<pre>{ "IsAvailable": true/false, "Description": "", "ErrorCode": 0, "ErrorMessage": "" }</pre>

سرویس های حوزه گواهی (Certificate Services)

این سرویس دارای هفت API است:

CertificateRequest

برای دریافت گواهی باید ابتدا اطلاعات هویتی متقاضی گواهی را در مرکز ثبت و شناسه یکتا دریافت کنید. برای این منظور از متد زیر استفاده می شود.

CertificateRequest API	
Request	
Method	Post
Body Content-Type	application/json
Encoding	UTF8
Header	CustomerCode= لایسنس- کد مشتری Signature= SignString(body,yourCertificate)
Body	<pre>{ "caName": "نام مرکز صدور", "profileName": "نام پروفایل گواهی", "signature": "SignString(customercode+"-"+licenseNumber+"-"+NationalCode, yourCertificate)", "requesterData": { "NationalCode": "شماره ملی متقاضی", "PostalCode": "کدپستی متقاضی", "Telephone": "شماره موبایل متقاضی", "BirthDate": "yyyy/mm/dd", "تاریخ تولد متقاضی گواهی به شمسی یا میلادی", "Email": "سایت مشتری @کدملی متقاضی) درج گردد, ایمیل متقاضی", "NationalCardSerialNo": "شماره سریال کارت ملی متقاضی", }, "eKYCMethod": "نوع احراز هویت", "eKYCCallback": "آدرس برگشت نتیجه احراز هویت", }</pre> <p>^۱ اختیاری ^۲ اختیاری ^۳ اختیاری ^۴ اختیاری</p>
Response	
Status Code	200 Success
Body	<pre>{ "IsSuccess" = true/false, "CertId" = long, شناسه یکتای درخواست گواهی "TrackingCode" = شناسه یکتا به ازاء هویت متقاضی, "کد رهگیری" "AdditionalData" = { "FirstName" = "نام", "LastName" = "نام خانوادگی", "EnFirstName" = "نام به لاتین", "EnLastName" = "نام خانوادگی به لاتین", "FatherName" = "نام پدر", "City" = "شهر", "ProvinceName" = "استان", "ProvinceID" = کد استان, "Address" = "آدرس", "AgencySN" = "شماره حرفه" }, "CertificateInfo" = { CSR = "", Certificate = "" } }</pre> <p>^۷ توجه</p>

```

    توجه۵
    "eKYCAuthenticated": true/false,
    "eKYCData" = {
        "orderId": "شناسه یکنای احراز هویت"
        "sign": "امضای توکن احراز هویت"
        "jwt": "توکن احراز هویت"
        "eKYC_Web": "آدرس وب احراز هویت"
        "eKYC_API": "آدرس سرویس احراز هویت"
        "eKYC_RedirectURL": "لینک احراز هویت تحت وب"
        "CallerCode": "کد مشتری در سامانه احراز هویت"
    },
    "ErrorCode" = "کد خطا",
    "ErrorMessage" = "متن خطا"
}
    
```

جدول نوع احراز هویت

کد	توضیح
۰	احراز هویت پندار را نمی خوام
۱	احراز هویت به روش وب Web redirect
۲	احراز هویت از طریق SDK
۳	احراز هویت از طریق سرویس های پایه Micro service

توجه داشته باشید :

- ۱- درج کدپستی فقط در مرکز میانی دولتی عام اجباری است و در سایر مراکز میانی در این موارد اختیاری می باشد.
- ۲- اگر از احراز هویت شرکت پندار کوشک ایمن برای احراز هویت استفاده نمی کنید باید مقدار eKYCMethod را صفر دهید در این حالت درج شماره سریال کارت ملی (NationalCardSerialNo) اختیاری است.
- ۳- درج آدرس برگشت (eKYCCallback) احراز هویت در صورت انتخاب نوع احراز هویت ۱ و ۲ (احراز هویت وب و SDK) اجباری است و در سایر موارد نیاز به درج مقدار برای eKYCCallback نمی باشد.
در آدرس برگشت مقادیر {token} و {tokenSignature} و {status} جایگزین می شود. نمونه آدرس:
<https://yourdomain/callback? token={token}&tokenSignature={tokenSignature}&status={status}>
برای شناخت مقادیر {token} و {tokenSignature} و مقادیر صحیح {status} به سند سرویس احراز هویت مراجعه کنید
- نکته: مقادیر ۵ و ۷ و ۸ و ۱۰ برای status به معنای عملیات موفق است.
- ۴- چنانچه این درخواست در گذشته احراز هویت شده باشد مقدار eKYCAuthenticated برابر true می باشد و اطلاعات eKYCData خالی خواهد بود و نیازی به احراز هویت متقاضی گواهی نمی باشد.
- ۵- کد رهگیری (TrackingCode) به ازای هر SubjectDN یکتا می باشد. یعنی برای یک فرد با اطلاعات ثابت همیشه کد رهگیری ثابت ارائه می شود. (این موضوع در مرکز میانی دولتی عام استثنا است)
- ۶- در صورتی که برای فرد متقاضی، گواهی فعال وجود داشته باشد در خروجی certificate و csr در CertificateInfo مقدار دهی می شوند. چنانچه کلید خصوصی متناظر با این گواهی یا CSR را دارید دیگر نیازی به انجام مرحله صدور گواهی نیست و از همین گواهی می توانید استفاده کنید و برای آن هزینه ای هم دریافت نمی شد. اما چنانچه کلید خصوص معادل گواهینامه را در اختیار ندارید باید حتما گواهی فعال فعلی را باطل کرده و مجدد فرآیند درخواست گواهی را از ابتدا انجام دهید.
- ۷- چنانچه کاربر گواهی فعال داشته باشد باید حتما گواهی خود را باطل کند در این حالت کد خطای ۱۰۰۱ دریافت خواهید کرد و برای سهولت در ابطال گواهی قبلی، گواهی فعال کاربر در متغیر Certificate در CertificateInfo باز گردانده می شود. تا در صورت تمایل به ابطال آن بتوانید به راحتی گواهی قبلی را باطل نمایید.
- ۸- جهت آشنایی با نحوه ارائه Email بند ۱۰ متد KeyStoreRequest را ببینید.

CertificateIssue

این متد جهت صدور گواهی مبتنی بر CSR استفاده می شود. برای استفاده از این متد نیاز به شناسه یکتای درخواست گواهی که از متد CertificateRequest بدست آماده می باشد.

CertificateIssue API	
Request	
Method	Post
Body Content-Type	application/json
Encoding	UTF8
Header	لایسنس- کد مشتری CustomerCode= Signature= SignString(body,yourCertificate)
Body	<pre>{ "certId": "شناسه یکتای درخواست گواهی", "csr": "Base64(CSR)", "signature": "SignBytes (CSR,yourCertificate)", "paymentId": "شناسه پرداخت" }</pre>
Response	
Status Code	200 Success
Body	<pre>{ "IsSuccess" = true/false, "Certificate" = Base64(Certificate), "CN" = Certificate CN, "Subject" = Certificate Subject, "IssuerName" = Certificate Issuer, "ValidFrom" = Certificate NotBefore, "ValidTo" = Certificate .NotAfter, "ErrorCode": "کد خطا", "ErrorMessage": "متن خطا", "Description" = "Error Description" }</pre>

توجه داشته باشید:

- اطلاعات هویتی که در CSR درج می شود باید دقیقا با اطلاعاتی که در زمان CertificateRequest در خروجی به شما داده می شود تطبیق داشته باشد در غیر این صورت خطای ۱۱۱۹ را دریافت خواهید کرد.
- ذاتا باینری می باشد و شما باید مقدار باینری را به Base64 تبدیل کرده و در پارامتر csr قرار دهید. اگر CSR تولیدی شما باینری نبوده و مستقیما Base64 تولید می شود (مثل خروجی SDK امضا در موبایل) باید همان مقدار را بدون تغییر در پارامتر csr بگذارید.
- مقدار ErrorMessage متن فارسی از خطای رخ داده است و در برنامه کاربردی می توان این متن را نمایش داد. از آنجا که این متن ممکن است بدون اطلاع تغییر کن هیچ تصمیم گیری بر اساس محتوای این متن نباید در برنامه و منطق آن صورت پذیرد و باید از مقدار ErrorCode برای فرآیندهای داخل برنامه استفاده نمود.
- مقدار Description حاوی متن کامل خطایی که رخ داده است می باشد و کاربرد آن برای برنامه نویسی و لاگ اطلاعات و رفع خطا است و به هیچ عنوان نباید در برنامه کاربردی نمایش داده شود.

KeyStoreRequest

این متد جهت دریافت گواهی حاوی کلید خصوصی در قالب استاندارد PKCS12 استفاده می‌شود در این نوع گواهی نیاز به تولید CRS نیست. این متد دقیقاً مثل متد CertificateRequest است و تنها تفاوت آن با متد قبلی این است که گواهی که به این ترتیب صادر می‌شود حاوی کلید خصوصی است و رمز آن نیز باید در درخواست ارسال شود.

KeystoreRequest API	
Request	
Method	Post
Body Content-Type	application/json
Encoding	UTF8
Header	CustomerCode= کدمشتری-لایسنس Signature= SignString(body,yourCertificate)
Body	<pre>{ "caName": "نام مرکز صدور", "profileName": "نام پروفایل گواهی", "signature": "SignString(customercode+"-" +licenseNumber+"-"+NationalCode, yourCertificate)", "requesterData": { "NationalCode": "شماره ملی متقاضی", "PostalCode": "کدپستی متقاضی", "Telephone": "شماره موبایل متقاضی", "BirthDate": "تاریخ تولد متقاضی", "Email": "پیشنهاد میشود جهت ایجاد هویت یکتا این مقدار (سایت مشتری @کدملی متقاضی) درج گردد, ایمیل متقاضی", "NationalCardSerialNo": "شماره سریال کارت ملی متقاضی", "Password": "حداکثر ۵۰ حرف" }, "eKYCMethod": "نوع احراز هویت", "eKYCCallback": "آدرس برگشت نتیجه احراز هویت" }</pre> <p>اختیاری^۱ اختیاری^۲ اختیاری^۳ اختیاری^۴</p>
Response	
Status Code	200 Success
Body	<pre>{ "IsSuccess" = true/false, "CertId" = long, شناسه یکتای درخواست گواهی "TrackingCode" = "کدرهگیری", "AdditionalData" = { "FirstName" = "نام", "LastName" = "نام خانوادگی", "EnFirstName" = "نام به لاتین", "EnLastName" = "نام خانوادگی به لاتین", "FatherName" = "نام پدر", "City" = "شهر", "ProvinceName" = "استان", "ProvinceID" = کد استان, "Address" = "آدرس", "AgencySN" = "شماره حرفه" }, "CertificateInfo" = { CSR = "", گواهی قبلی کاربر در صورت وجود Certificate = "", گواهی قبلی کاربر در صورت وجود }, "eKYCAuthenticated": true/false, "eKYCData" = { "orderId": "شناسه یکتای احراز هویت" } }</pre> <p>توجه^۷ توجه^۵</p>

```

"امضای توکن احراز هویت": "sign"
"توکن احراز هویت": "jwt"
"آدرس وب احراز هویت": "eKYC_Web"
"آدرس سرویس احراز هویت": "eKYC_API"
"لینک احراز هویت تحت وب": "eKYC_RedirectURL"
"کد مشتری در سامانه احراز هویت": "CallerCode"
},
"ErrorCode" = "کد خطا",
"ErrorMessage" = "متن خطا"
}
    
```

جدول نوع احراز هویت

کد	توضیح
۰	احراز هویت پندار را نمی‌خوام
۱	احراز هویت به روش وب Web redirect
۲	احراز هویت از طریق SDK
۳	احراز هویت از طریق سوریس‌های پایه Micro service

توجه داشته باشید :

- ۱- درج کدپستی فقط در مرکز میانی دولتی عام اجباری است و در سایر مراکز میانی در این موارد اختیاری می‌باشد.
- ۲- اگر از احراز هویت شرکت پندار کوشک ایمن برای احراز هویت استفاده نمی‌کنید باید مقدار eKYCMethod را صفر دهید در این حالت درج شماره سریال کارت ملی (NationalCardSerialNo) اختیاری است.
- ۳- درج آدرس برگشت (eKYCCallback) احراز هویت در صورت انتخاب نوع احراز هویت ۱ و ۲ (احراز هویت وب و SDK) اجباری است و در سایر موارد نیاز به درج مقدار برای eKYCCallback نمی‌باشد.
در آدرس برگشت مقادیر {token} و {tokenSignature} و {status} جایگزین می‌شود. نمونه آدرس:
`https://yourdomain/callback? token={token}&tokenSignature={tokenSignature}&status={status}`
برای شناخت مقادیر {token} و {tokenSignature} و مقادیر صحیح {status} به سند سرویس احراز هویت مراجعه کنید
- نکته: مقادیر ۵ و ۷ و ۸ و ۱۰ برای status به معنای عملیات موفق است.
- ۴- درج آدرس برگشت (eKYCCallback) احراز هویت در صورت انتخاب نوع احراز هویت ۱ و ۲ (احراز هویت وب و SDK) اجباری است و در سایر موارد نیاز به درج مقدار برای eKYCCallback نمی‌باشد.
- ۵- چنانچه این درخواست در گذشته احراز هویت شده باشد مقدار eKYCAuthenticated برابر true می‌باشد و اطلاعات eKYCData خالی خواهد بود و نیازی به احراز هویت متقاضی گواهی نمی‌باشد.
- ۶- کد رهگیری (TrackingCode) به ازای هر SubjectDN یکتا می‌باشد. یعنی برای یک فرد با اطلاعات ثابت همیشه کد رهگیری ثابت ارائه می‌شود. (این موضوع در مرکز میانی دولتی عام استثنا است)
- ۷- در صورتی که برای فرد متقاضی، گواهی فعال وجود داشته باشد در خروجی certificate و csr در CerificateInfo مقدار دهی می‌شوند. چنانچه کلید خصوصی متناظر با این گواهی یا CSR را دارید دیگر نیازی به انجام مرحله صدور گواهی نیست و از همین گواهی می‌توانید استفاده کنید و برای آن هزینه ای هم دریافت نمی‌شد. اما چنانچه کلید خصوص معادل گواهینامه را در اختیار ندارید باید حتما گواهی فعال فعلی را باطل کرده و مجدد فرآیند درخواست گواهی را از ابتدا انجام دهید.



- ۸- چنانچه کاربر گواهی فعال داشته باشد باید حتما گواهی خود را باطل کند در این حالت کد خطای ۱۰۰۱ دریافت خواهید کرد و برای سهولت در ابطال گواهی قبلی، گواهی فعال کاربر در متغیر Certificate در CertificateInfo باز گردانده می‌شود. تا در صورت تمایل به ابطال آن بتوانید به راحتی گواهی قبلی را باطل نمایید.
- ۹- مقدار password می‌تواند بصورت رمز شده ارسال شود برای این کار باید پسورد انتخابی با کدینگ UTF8 تبدیل به باید شده و سپس با گواهی برنامه RA که در آدرس زیر قرار دارد و پدینگ PKCS1 رمز شود و نتیجه آن در قالب Base64 در پارامتر password قرار گیرد.

<https://pki.co.ir/download/PRA/PKIRAEncryptCertificate.cer>

نمونه کد برای رمزنگاری پسورد به زبان C#:

```
public static string EncryptDataByRACert(string password)
{
    X509Certificate2 certificate = new X509Certificate2( /*گواهی رمزنگاری گذرگاه*/);
    RSA rsa = certificate.GetRSAPublicKey();
    byte[] data = Encoding.UTF8.GetBytes(password);
    byte[] encryptedPass = rsa.Encrypt(data, RSAEncryptionPadding.Pkcs1);
    return Convert.ToBase64String(encryptedPass);
}
```

گواهی رمزنگاری گذرگاه در آدرس زیر می‌باشد :

<https://pki.co.ir/download/PRA/PKIRAEncryptCertificate.cer>

پیشنهاد می‌شود برای حفظ امنیت پسورد، پسورد حتما به صورت رمز شده ارسال گردد. لازم به ذکر است این رمز نه توسط سرویس گذرگاه و نه توسط CA نمی‌شود و در صورت فراموشی گواهی غیر قابل استفاده خواهد بود و صاحب آن باید گواهی را ابطال و گواهی جدید با پرداخت هزینه دریافت کند.

۱۰- مقدار Email اختیاری است اما پیشنهاد می‌شود این پارامتر با مقداری با قالب زیر پر شود. این امر باعث می‌شود در Subject گواهی نام دامنه شما قرار گیرد و در نتیجه گواهی برای شما صادر و یکتا گردد. در این حالت چنانچه متقاضی گواهی از جای دیگری گواهی داشته باشد باز هم می‌تواند نزد شما گواهی بگیرد.

Email = کدملی متقاضی گواهی @yoursite

بعنوان مثال : 1234567890@pki.co.ir

دقت کنید این آدرس ایمیل باید حتما در CSR تولیدی نیز با مقدار درج شود.

KeyStoreIssue

این متد جهت صدور گواهی مبتنی بر PKCS12 حاوی کلید خصوصی استفاده می‌شود. برای استفاده از این متد نیاز به شناسه یکتای درخواست گواهی که از متد KeyStoreRequest بدست آماده می‌باشد.

KeystoreIssue API	
Request	
Method	Post
Body Content-Type	application/json
Encoding	UTF8
Header	لایسنس- کد مشتری CustomerCode= Signature= SignString(body, yourCertificate)
Body	<pre>{ "certId": "شناسه یکتای درخواست گواهی", "password": "حداکثر ۵۰ حرف", "signature": "SignString (password, yourCertificate)", "paymentId": "شناسه پرداخت" }</pre>
Response	
Status Code	200 Success
Body	<pre>{ "IsSuccess" = true/false, "Certificate" = Base64(Certificate), "KeyStore" = Base64(KeyStore P12 format), "CN" = Certificate CN, "Subject" = Certificate Subject, "IssuerName" = Certificate Issuer, "ValidFrom" = Certificate NotBefore, "ValidTo" = Certificate .NotAfter, "ErrorCode": "کد خطا", "ErrorMessage": "متن خطا", "Description" = "Error Description" }</pre>

توجه داشته باشید :

- ۱- رمز عبور باید دقیقاً همان رمزی باشد که در متد KeyStoreRequest داده شده بود.
- ۲- مرکز صدور رمز گواهی را نگهداری نمی‌کند و در صورت گم شدن آن باید گواهی باطل شده و گواهی جدید گرفته شود.
- ۳- مقدار ErrorMessage متن فارسی از خطای رخ داده است و در برنامه کاربردی می‌توان این متن را نمایش داد. از آنجا که این متن ممکن است بدون اطلاع تغییر کن هیچ تصمیم‌گیری بر اساس محتوای این متن نباید در برنامه و منطق آن صورت پذیرد و باید از مقدار ErrorCode برای فرآیندهای داخل برنامه استفاده نمود.
- ۴- مقدار Description حاوی متن کامل خطایی که رخ داده است می‌باشد و کاربرد آن برای برنامه نویسی و لاگ اطلاعات و رفع خطا است و به هیچ عنوان نباید در برنامه کاربردی نمایش داده شود.
- ۵- پیشنهاد می‌شود مقدار password بصورت رمز شده ارسال شود. برای این موضوع به بند ۹ متد KeyStoreRequest مراجعه کنید.

RevokeCertificate

این متد جهت ابطال گواهی می باشد:

RevokeCertificate API	
Request	
Method	Post
Body Content-Type	application/json
Encoding	UTF8
Header	لایسنس - کد مشتری = CustomerCode Signature= SignString(body,yourCertificate)
Body	<pre>{ "certificate" : "base64(certificate)", "signature" : "SignBytes (certificate, yourCertificate))", "revokeRequestLetter": "متن درخواست ابطال", "revokeCertReason" : 3 }</pre>
Response	
Status Code	200 Success
200 Schema	Boolean
	<pre>{ "IsSuccess" : true/false, "Description" : revokeDescription, "ErrorCode": "کد خطا", "ErrorMessage" : "متن خطا" }</pre>

جدول دلیل ابطال گواهی مطابق استاندارد مرکز دولتی ریشه

کد	توضیح
۰	AffiliationChanged
۱	KeyCompromise
۲	PrivilegesWithdrawn
۳	Unspecified

جدول دلیل ابطال گواهی مطابق با استاندارد RFC 5280

<https://datatracker.ietf.org/doc/html/rfc5280#section->

(5.3.1)

کد	توضیح
۱	keyCompromise
۲	cACompromise
۳	affiliationChanged
۴	superseded
۵	cessationOfOperation
۶	certificateHold
۹	privilegeWithdrawn
۱۰	aACompromise

توجه : Certificate ذاتا باینری می باشد و شما باید مقدار باینری را به Base64 تبدیل کرده و در پارامتر certificate قرار دهید. اگر Certificate تولیدی شما باینری نبوده و مستقیما Base64 تولید می شود باید همان مقدار را بدون تغییر در پارامتر certificate بگذارید.



AuthenticationCompleted

پس از انجام عملیات احراز هویت این متد باید اجرا شود تا مرکز صدور از انجام عملیات احراز هویت اطمینان حاصل کند. لازم به ذکر این متد چه در صورت استفاده از احراز هویت شرکت پندار کوشک ایمن چه احراز هویت دیگر باید حتما اجرا گردد.

چنانچه این متد اجرا نشود امکان صدور گواهی وجود نخواهد داشت.

AuthenticationCompleted API	
Request	
Method	Post
Body Content-Type	application/json
Encoding	UTF8
Header	CustomerCode= لایسنس - کد مشتری Signature= SignString(body,yourCertificate)
Body	{ "certId": "شناسه یکتای درخواست گواهی", "orderId": "شناسه احراز هویت", "authenticator": "نام احراز هویت" }
Response	
Status Code	200 Success
200 Schema	Boolean
	{ "IsSuccess" = true/false, "Description" = توضیح خطا, "ErrorCode": کد خطا, "ErrorMessage": "متن خطا" }

توجه:

۱- چنانچه در مرحله درخواست certificateRequest/keystoreRequest اعلام کرده باشید که احراز هویت نمی خواهید. یعنی مقدار eKYCMethod را برابر صفر داده باشید در این متد مقدار orderId باید حتما خالی باشد.



RecievedCertConfirmation

پس از اینکه گواهی با موفقیت صادر شد با کمک این متد می توان به مرکز صدور اعلام کرد که گواهی توسط متقاضی دریافت شده است. اجرای این متد الزامی نیست اما در تایید نهایی اطلاعات بسیار مفید خواهد بود.

RecievedCertConfirmation API	
Request	
Method	Post
Body Content-Type	application/json
Encoding	UTF8
Header	لایسنس - کد مشتری CustomerCode= Signature= SignString(body,yourCertificate)
Body	{ "certId": شناسه یکتای درخواست گواهی, }
Response	
Status Code	200 Success
200 Schema	Boolean
	{ "IsSuccess" = true/false, "Description" = "" , "ErrorCode": 0, "ErrorMessage": "" }



سرویس های گزارشگیری (Reporting Services)

این سرویس دارای هفت API است:

IssuingReport

این متد جهت دریافت گزارش آمار گواهی های ثبت شده، صادر شده و یا باطل شده می باشد:

IssuingReport API	
Request	
Method	Post
Body Content-Type	application/json
Encoding	UTF8
Header	CustomerCode= لایسنس - کد مشتری Signature= SignString(body,yourCertificate)
Body	{ "startdate": "yyyy/mm/dd", "enddate": "yyyy/mm/dd", "type": "PerCA/PerCA&Status/detail", "certId": "ردیف گواهی" }
Response	
Status Code	200 Success
200 Schema	Boolean
	گزارش گواهی های صادر شده به تفکیک مرکز صدور در حالت detail به دلیل پاسخدهی سریع سرویس فقط ۱۰۰ مورد بر می گردد و برای دریافت اطلاعات بیشتر سرویس باید با certId آخر مرحله قبل مجدد فراخوانی شود



IsRequestAuthenticated

این متد جهت دریافت وضعیت احراز هویت یک درخواست گواهی می باشد:

IsRequestAuthenticated API	
Request	
Method	Post
Body Content-Type	application/json
Encoding	UTF8
Header	CustomerCode= لایسنس - کد مشتری Signature= Base64(Sign_RSAWithSHA1(UTF8.Byte(body)))
Body	{ "certId": "ردیف گواهی" }
Response	
Status Code	200 Success
200 Schema	Boolean
	{ "IsSuccess" = true/false, "CertificateAuthenticated" = true/false , "ErrorCode": 0, "ErrorMessage": "" }

چنانچه در خواست احراز هویت شده باشد مقدار CertificateAuthenticated برابر true خواهد بود.

بدیهی است در صورتی که اطلاعات یک درخواست احراز هویت شده، تغییر یاب احراز هویت آن باطل خواهد شد.

پیوست شماره ۱

شرح خطا	کد خطا
اجرای موفق	۰
خطا در اجرای برنامه	۱
عملیات ناموفق	۲
عدم پشتیبانی از دستور	۳
شما با این اطلاعات گواهینامه فعال دارید برای دریافت گواهی ابتدا باید آن را باطل کنید	۱۰۰۱
خطای ناشناخته در مرحله بروزرسانی اطلاعات	۱۰۰۲
نام وارد شده با کد ملی انطباق ندارد	۱۰۰۳
نام خانوادگی وارد شده با کد ملی انطباق ندارد	۱۰۰۴
مرکز میانی نماد قطع است	۱۰۰۵
مرکز صدور یا پروفایل انتخاب شده نامعتبر است	۱۱۰۰
مرکز صدور گواهینامه نامعتبر است	۱۱۰۱
شما به این مرکز صدور دسترسی ندارید	۱۱۰۲
درج کد ملی الزامی است	۱۱۰۳
درج شماره تلفن همراه الزامی است	۱۱۰۴
امضا کننده نامه درخواست گواهینامه مجاز به امضای این نامه نمی باشد	۱۱۰۵
امضای نامه درخواست گواهینامه نامعتبر است	۱۱۰۶
گواهی امضا کننده نامه درخواست باطل شده است	۱۱۰۷
گواهی امضا کننده نامه درخواست ناشناخته است	۱۱۰۸
در اعتبارسنجی گواهی امضا کننده نامه درخواست خطا رخ داد	۱۱۰۹
کد رهگیری احراز هویت نشده است	۱۱۱۰
“کد رهگیری نامعتبر است	۱۱۱۱
گواهی برای ابطال یافت نشد	۱۱۱۲
این گواهی توسط شما صادر نشده و حق ابطال آن را ندارید	۱۱۱۳
این گواهی قبلا باطل شده است	۱۱۱۴
این فرد در قید حیات نیست	۱۱۱۵
شماره موبایل در مالکیت شما نیست و یا سرویس شاهکار قطع است	۱۱۱۶
سرویس ثبت احوال قطع است	۱۱۱۷
سرویس شاهکار قطع است	۱۱۱۸
اطلاعات درخواست گواهی با اطلاعات ثبتی انطباق ندارد	۱۱۱۹
کد ملی با کد رهگیری منطبق نمی باشد	۱۱۲۰

پرداخت کننده و متقاضی یکسان نمی باشند	۱۱۲۱
امضای بسته نامعتبر است	۱۲۰۰
کد پرداخت نامعتبر است	۱۲۰۱
مبلغ پرداختی با هزینه گواهی انطباق ندارد	۱۲۰۲
کد مشتری نامعتبر است	۱۲۰۳
اطلاعاتی یافت نشد	۱۲۰۴
کد سفارش با کد پرداخت انطباق ندارد	۱۲۰۵
این سفارش پرداخت نشده است	۱۲۰۶
درج تاریخ تولد الزامی است	۱۲۰۷
این فرد احراز هویت نشده است	۱۲۰۸
پرداخت با موفقیت انجام نشد	۱۲۰۹
کد تسهیم نامعتبر است	۱۲۱۰
توکن الزامی است	۱۲۱۱
کد رهگیری قبلا تایید شده است	۱۲۱۲
این درخواست باید بصورت حضوری احراز هویت شود	۱۲۱۳
پسورد الزامی است	۱۲۱۴



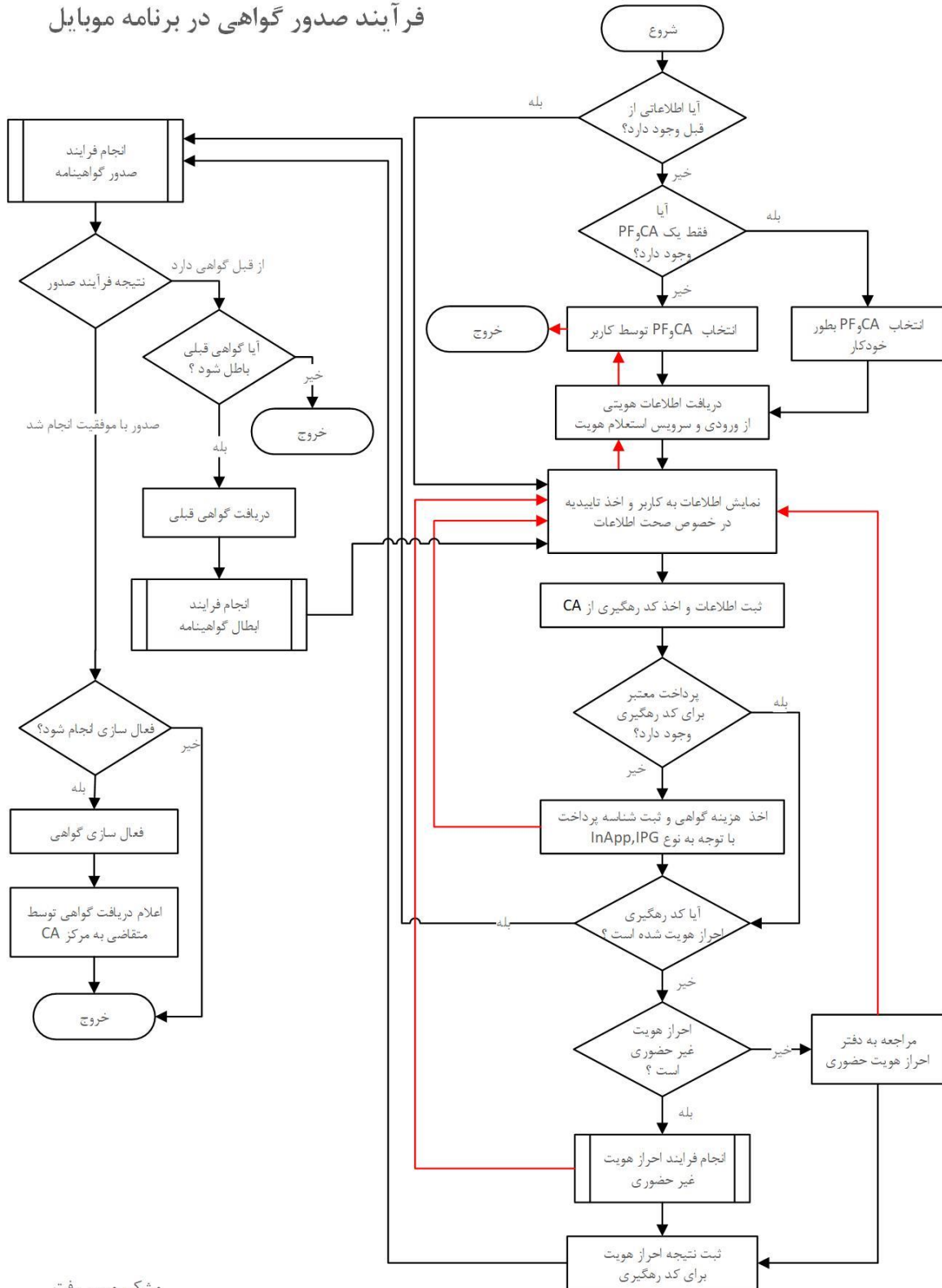
پیوست شماره ۲

اطلاعات مورد نیاز جهت تعریف کد مشتری به شرح زیر می باشد. این اطلاعات به همراه کلید عمومی مشتری باید به آدرس ایمیل sales@pki.co.ir با عنوان “درخواست ایجاد کد مشتری در سرویس صدور گواهی” یا به رابط فروش خود در شرکت ارسال شود.

- ۱- نام شرکت
- ۲- شناسه ملی شرکت
- ۳- کدپستی و آدرس شرکت
- ۴- آدرس سایت شرکت
- ۵- آدرس ایمیل شرکت
- ۶- تلفن شرکت
- ۷- نام و نام خانوادگی مدیرعامل
- ۸- شماره ملی مدیرعامل
- ۹- تلفن همراه مدیرعامل
- ۱۰- گواهی الکترونیکی یا کلید عمومی با طول کلید ۱۰۲۴ الگوریتم RSA اختصاصی شرکت

پیوست شماره ۳

فرآیند صدور گواهی در برنامه موبایل



مشکی مسیر رفت
قرمز مسیر برگشت



پیوست شماره ۴

نحوه تولید زوج کلید در تصدیق هویت متقاضی

همانگونه که در بخش سرویس های بیان گردید، برای هر تراکنش نیاز به تصدیق هویت متقاضی از طریق امضای اختصاصی وی (RSA) (Sign) در آن تراکنش می باشد. جهت راهنمایی بهتر متقاضیان سعی شده که در این پیوست با استفاده از OpenSSL اقدام به ایجاد زوج کلید تصدیق هویت شود.

در ابتدا لازمست تا نرم افزار openssl نسخه ویندوز خود را دانلود کنید:

<https://slproweb.com/products/Win32OpenSSL.html>

در مسیر برنامه، دستور زیر را اجرا کنید. توجه کنید که بعد از O نام شرکت و بعد از CN نام خود و یا نام پروژه را قرار دهید:

```
openssl req -x509 -newkey rsa:1024 -keyout key.pem -out cert.pem -sha256 -days 3650 -nodes -subj "/C=IR/O=Company_Name/CN=Developer_or_Project_Name/OU=Department"
```

در صورت اجرای موفق دستور فوق، یک فایل بنام cert.pem حاوی کلید عمومی ایجاد می شود که باید آنرا برای شرکت ارسال کنید.

با دستور زیر و به منظور نگهداری امن کلیدها، می توانید فایل ها را به قالب pfx تبدیل کنید:

```
openssl pkcs12 -inkey key.pem -in cert.pem -export -out Cert.pfx
```

با این اقدام، می توان دو فایل pem را حذف نمود.