

# سرویس‌های درگاه اینترنتی امضا

راهنمای بهره‌برداران



واحد امنیت اطلاعات و زیرساخت کلید عمومی

نسخه	۲.۴.۹
تاریخ	دی ماه ۱۴۰۲
شناسه	PKI-ISG-API-DG
طبقه بندی	عمومی

## تاریخچه

نسخه	تاریخ	تهیه کنندگان	مرور کنندگان	توضیحات
۲.۰.۰	۱۴۰۰/۱۱/۰۲	واحد امنیت	واحد کنترل کیفیت	تهیه سند
۲.۰.۱	۱۴۰۰/۱۱/۰۵	واحد امنیت	واحد کنترل کیفیت	تکمیل توضیحات سند
۲.۰.۲	۱۴۰۰/۱۱/۰۷	واحد امنیت	واحد کنترل کیفیت	تکمیل توضیحات API های: GetSigningToken ResendNotiication
۲.۲.۰	۱۴۰۰/۱۱/۲۵	واحد امنیت	واحد کنترل کیفیت	اضافه شدن HandwriteSignature و ContentImage
۲.۲.۱	۱۴۰۱/۰۱/۳۰	واحد امنیت	واحد کنترل کیفیت	تکمیل توضیحات HandwriteSignature و ContentImage
۲.۲.۲	۱۴۰۱/۰۳/۰۴	واحد امنیت	واحد کنترل کیفیت	نکته قابل توجه در چک کردن base64string
۲.۳.۰	۱۴۰۱/۰۴/۲۹	واحد امنیت	واحد کنترل کیفیت	به روز رسانی سند و امضای نوع CMS Attached
۲.۳.۱	۱۴۰۱/۰۵/۰۴	واحد امنیت	واحد کنترل کیفیت	تکمیل توضیحات ContentImage
۲.۴.۰	۱۴۰۱/۰۷/۳۰	واحد امنیت	واحد کنترل کیفیت	بهبود صف امضا
۲.۴.۲	۱۴۰۱/۰۸/۰۷	واحد امنیت	واحد کنترل کیفیت	تبدیل نسخه PDF
۲.۴.۳	۱۴۰۱/۰۸/۳۰	واحد امنیت	واحد کنترل کیفیت	محدودیت تاریخ انقضای محتوا به ۱۰ روز
۲.۴.۴	۱۴۰۱/۰۹/۰۵	واحد امنیت	واحد کنترل کیفیت	- افزودن قابلیت رمز گذاری و رمز گشایی محتوای امضا - کنترل حجم محتوای امضا با توجه به حجم تنظیم شده در Config - محدودیت تاریخ انقضای امضا با توجه زمان تنظیم شده در Config
۲.۴.۵	۱۴۰۱/۰۹/۲۹	واحد امنیت	واحد کنترل کیفیت	یکسان کردن Conversion های محتوای امضا افزودن Certificate Validation به نمودار امضا
۲.۴.۶	۱۴۰۱/۱۰/۰۹	واحد امنیت	واحد کنترل کیفیت	توضیحات بیشتر در مورد برخی فیلدهای GetSigningToken
۲.۴.۷	۱۴۰۲/۰۶/۰۸	واحد امنیت	واحد کنترل کیفیت	افزودن قابلیت عدم اعتبارسنجی گواهینامه صادر شده از مراکز میانی که قابلیت اعتبارسنجی گواهی را ندارند.
۲.۴.۸	۱۴۰۲/۰۷/۰۱	واحد امنیت	واحد کنترل کیفیت	افزودن تایپ چکاد
۲.۴.۹	۱۴۰۲/۱۰/۰۵	واحد امنیت	واحد کنترل کیفیت	تغییر خروجی ارسال نوتیفیکیشن و افزودن توابع ذیل: GetSigningList RemoveFromQueue

## پیوست‌ها

ردیف	نام	نسخه	توضیحات
۱	ISG.DTO.nupkg	1.6.9	Binary and source code
۲	ISG.DTO.jar	1.6.9	Binary and source code
۳	ISG.Demo	2.5.4	Sample Application with source code

## فهرست مطالب

۲	تاریخچه
۳	پیوست‌ها
۴	فهرست مطالب
۵	پیشگفتار
۶	فرآیند امضای Plain آنلاین
۷	فرآیند امضای رمزگذاری شده آنلاین
۸	سرویس‌های درگاه
۹	سرویس احراز هویت (Auth Service)
۹	Login API
۹	Logout API
۱۰	سرویس توکن احراز هویت (Token Service)
۱۰	Refresh
۱۱	Revoke
۱۱	سرویس امضا (Signing Service)
۱۱	ChangePassword
۱۱	AllTemplates
۱۲	TemplateInfo
۱۳	TemplateStructure
۱۴	GetSigningToken
۱۸	CheckStatus
۱۸	GetData
۲۰	RemoveData
۲۱	ResendNotification
۲۲	سرویس Notification (Notification Service)
۲۲	GetSigningQueueList
۲۲	RemoveFromQueue

۲۴	..... سرویس Tools (Tools Service)
۲۴	..... EncryptData
۲۶	..... شمای تبادل اطلاعات (Schemas)
۳۲	..... ضمیمه

## پیشگفتار

با توجه به شتاب توسعه فناوری اطلاعات در حوزه نرم افزارهای کاربردی و همچنین نیاز به تولید محصولات نرم افزاری در پلتفرم‌های متنوع، دیگر تولید SDK به تمام زبان‌های برنامه‌نویسی امری منطقی محسوب نمی‌گردد. از سوی دیگر ارائه نرم‌افزار به عنوان یک سرویس (SaaS) راهکاری است که در سالیان اخیر پاسخگوی بسیاری از نیازمندی‌های موسسات و سازمان‌های متوسط تا بزرگ در سطح جهان بوده است.

درگاه اینترنتی امضا ابزاری برای تعامل با حوزه زیرساخت کلید عمومی در تمامی پلتفرم‌ها است که می‌تواند نیازمندی‌های بهره‌برداران و دغدغه‌های برنامه‌نویسان را در این حوزه پوشش دهد.

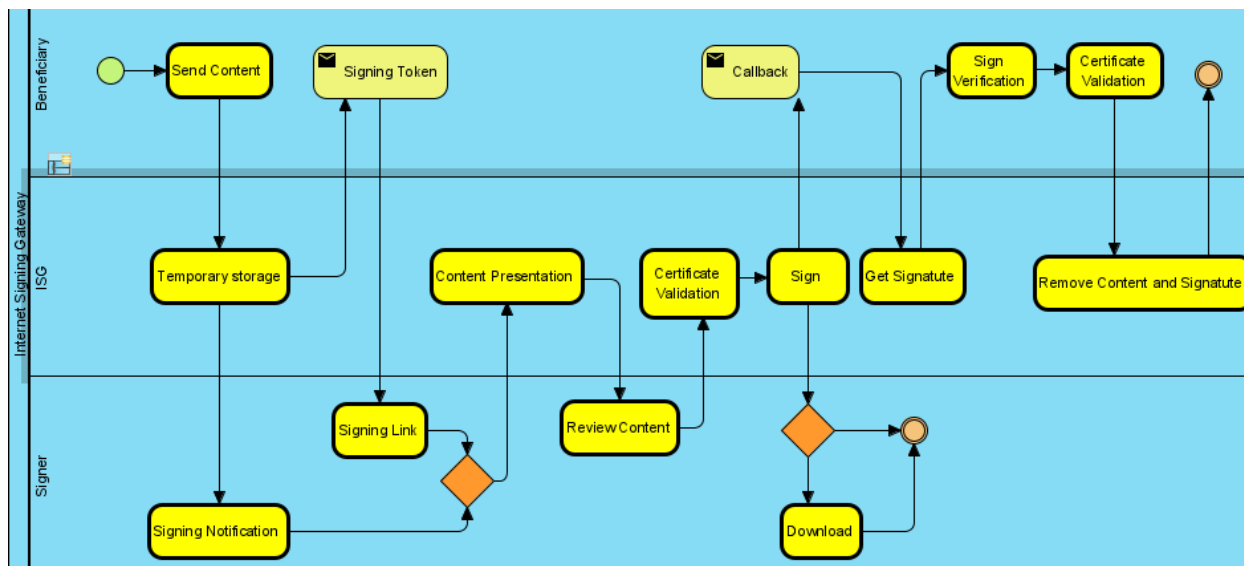
برای به دست آوردن دیدی روشن از درگاه امضا، می‌توان خریدهای اینترنتی را در نظر گرفت. بطور معمول هر گاه از یک فروشگاه آنلاین خریدی انجام شود، عملیات پرداخت وجه از درگاه پرداخت بانکی صورت می‌گیرد. در حقیقت پذیرنده یا همان فروشگاه آنلاین، به هیچ عنوان درگیر عملیات انتقال وجه نبوده و تنها پس از پایان تعاملات لازم بین مشتری و بانک، رسید پرداخت را از بانک دریافت می‌نماید. درگاه اینترنتی امضا نیز مشابه همین رویکرد را دارد. پس از آماده شدن محتوایی که باید امضا شود، فرد امضاکننده به درگاه هدایت می‌شود. در آنجا مطابق قوانین و استانداردهای لازم، ضمن رویت محتوا، آنرا خوانده، بررسی نموده و در صورت موافقت، امضا خواهد نمود. آنگاه نتیجه امضا برای تهیه‌کننده محتوا ارسال خواهد شد.

در این سند ابتدا فرآیند امضای آنلاین بررسی شده، سپس سرویس‌های درگاه که تعاملات لازم در فرآیند را ارائه می‌دهند، توصیف خواهند شد.

در درگاه امضا، به دو صورت می‌توان فرآیند امضا را انجام داد: مدل Plain و مدل امضای رمزگذاری شده.

مدل Plain به صورت ذیل می‌باشد:

## فرآیند امضای Plain آنلاین

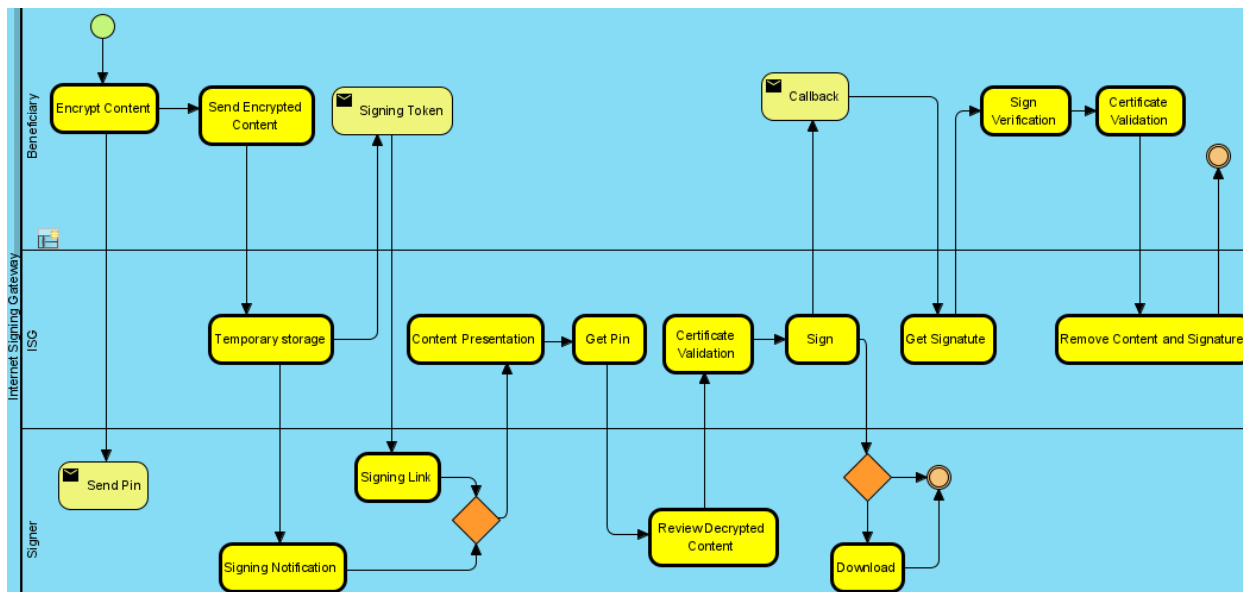


همانگونه که در نمودار فوق مشخص است، کلیات فرآیند Plain امضا چنین است:

۱. بهره‌بردار محتوای مورد نظر را برای درگاه امضا ارسال می‌نماید.
۲. درگاه محتوای دریافتی را به صورت موقت ذخیره می‌کند.
۳. درگاه دریافت محتوا را اطلاع می‌دهد:
- ۳/۱. در پاسخ به ارسال محتوا، یک شناسه ارجاع به محتوا به بهره‌بردار ارایه می‌دهد.
- ۳/۲. در صورت درخواست بهره‌بردار یک پیام اطلاع‌رسانی به گوشی همراه امضاکننده ارسال می‌نماید.
۴. متناسب با درخواست بهره‌بردار، امضاکننده یا توسط خود بهره‌بردار یا توسط پیام دریافتی از درگاه، به درگاه امضا هدایت می‌شود.
۵. درگاه محتوا را به امضاکننده ارایه می‌نماید.
۶. امضاکننده ضمن رویت و بررسی محتوا، با استفاده از امضای دیجیتال خود اقدام به امضای آن می‌نماید.
۷. درگاه، پس از اعتبارسنجی گواهی کاربر و تکمیل فرآیند امضا، امضا شدن محتوا را به بهره‌بردار اطلاع می‌دهد.
- ۷/۱. در صورتی که بهره‌بردار هنگام ارسال محتوا، اجازه دانلود توسط امضاکننده را داده باشد، درگاه امکان دانلود را در اختیار امضاکننده قرار می‌دهد.
- ۷/۲. تعامل امضاکننده با یا بدون دانلود محتوای امضا شده به پایان می‌رسد.
۸. بهره‌بردار امضا را از درگاه دریافت می‌نماید.
۹. بهره‌بردار، امضا دریافتی را اعتبارسنجی می‌نماید.

۱۰. اگر مرکز میانی گواهی، امکان اعتبارسنجی گواهی را داشته باشد، بهره‌بردار، باید گواهی کاربر را با استفاده از سرویس DSS یا SDK PKTB، اعتبارسنجی نماید. (در نمونه برنامه پیوست این سند به نام ISG.Demo، فراخوانی‌های مربوط به اعتبارسنجی گواهی با استفاده از DSS و SDK PKTB پیاده‌سازی شده و قابل بررسی برای توسعه‌دهندگان می‌باشد).
۱۱. بهره‌بردار، به درگاه اعلام می‌نماید که محتوا و امضای آن را حذف نماید.
۱۲. درگاه نتیجه حذف اطلاعات را به بهره‌بردار اعلام و فرآیند به پایان می‌رسد.

## فرآیند امضای رمزگذاری شده آنلاین



همانگونه که در نمودار فوق مشخص است، فرآیند امضای رمزگذاری شده مانند امضای Plain است با این تفاوت که در مرحله اول، بهره‌بردار باید محتوای موردنظر را به صورت رمزگذاری شده برای درگاه امضا ارسال نماید.

سایر مراحل تا مرحله ۵ نیز مانند فرآیند Plain انجام می‌شود.

در مرحله ۵، برای نمایش محتوای امضا، ابتدا پنجره‌ای به امضاکننده نمایش داده می‌شود تا رمزی که برای ایشان از طرف بهره‌بردار ارسال شده است را وارد نموده و در صورت ورود رمز صحیح، محتوای امضا نمایش داده می‌شود. در غیر این صورت محتوای امضا قابل نمایش نخواهد بود.

و در صورت ورود صحیح رمز و پس از نمایش محتوا، سایر مراحل نیز مانند فرآیند امضای Plain انجام می‌گردد.

شایان ذکر است که Encrypt محتوای امضا به صورت ذیل انجام می‌گردد:

بهره‌بردار باید برای هر عملیات امضا، یک رمز ایجاد کرده و این رمز را به همراه محتوایی که باید برای امضا ارسال شود، به تابع Encrypt که طی یک الگوریتم توافقی، عملیات رمزگذاری محتوا را انجام و نتیجه را به صورت یک رشته‌ی Base64 برمیگرداند،

ارسال نماید. پس از انجام عملیات Encrypt امضا، بهره‌بردار باید محتوای Encrypt شده را به درگاه امضا و رمز را نیز برای امضاکننده ارسال نماید و امضاکننده از این رمز، زمان نمایش محتوا برای رمزگشایی محتوا استفاده می‌نماید.

## سرویس‌های درگاه

جهت آغاز فرآیند امضای آنلاین و ادامه تعامل با درگاه سرویس‌هایی به شرح زیر ارائه شده‌اند:

### Auth

- ✚ [Login](#)
- ✚ [Logout](#)

### Token

- ✚ [Refresh](#)
- ✚ [Revoke](#)

### Signing

- ✚ [ChangePassword](#)
- ✚ [AllTemplates](#)
- ✚ [TemplateInfo](#)
- ✚ [TemplateStructure](#)
- ✚ [GetSigningToken](#)
- ✚ [CheckStatus](#)
- ✚ [GetData](#)
- ✚ [RemoveData](#)
- ✚ [ResendNotification](#)

### Notification

- ✚ [GetSigningQueueList](#)
- ✚ [RemoveFromQueue](#)

### Tools

- ✚ [EncryptData](#)

**نکته:** در این نسخه احراز هویت (Authentication) فراخواننده سرویس‌ها بر اساس نام کاربری و گذرواژه است. در نسخه آتی احراز هویت مبتنی بر `challenge – sign` خواهد بود. کنترل دسترسی (Authorization) نیز بر اساس JWT انجام می‌شود. در تمامی فراخوانی‌ها، مجاز بودن نشانی IP فراخواننده API متناسب با IP‌های معرفی شده در حساب کاربری بررسی می‌شود.



## سرویس احراز هویت (Auth Service)

### Login API

در گام اول تعامل با درگاه، لازم است احراز هویت انجام شود. با فراخوانی Login API و ارسال نام کاربری و گذرواژه، یک JWT دریافت می‌شود که در تعاملات آتی مورد استفاده قرار می‌گیرد. پارامترهای ارسالی به این API در کلاس `AuthRequest` توصیف شده‌اند. پاسخ دریافتی نیز مطابق کلاس `AuthResponse` است. پس از احراز هویت موفقیت‌آمیز و دریافت `accessToken` در فراخوانی‌های بعدی سایر API‌ها باید توکن دریافتی را در بخش `authorize` مربوط به `header` درخواست خود قرار دهید:

Authorization value: Bearer accessToken

تنظیمات پیش‌فرض API‌ها طول عمری معادل ۱۰ دقیقه برای `accessToken` در نظر می‌گیرد. پس از طی این مدت `accessToken` منقضی خواهد شد. پس از انقضا می‌توانید در طول روز جاری با فراخوانی API `Refresh` و ارسال همزمان `accessToken` و `refreshToken` توکن جدیدی دریافت نمایید که به طور مجدد، برای ۱۰ دقیقه قابل استفاده خواهد بود. طول عمر `accessToken` و `refreshToken` توسط مدیر سیستم قابل تنظیم است. مشخصات `request` ارسالی به این API و `response` دریافتی به شرح جدول زیر است:

Login API	
Request	
Method	Post
Body Content-Type	application/json
Schema	<code>AuthRequest</code>
	<pre>{   "userName": "string",   "password": "string" }</pre>
Response	
Status Code	200 Success / 401 Unauthorized
200 Schema	<code>AuthResponse</code>
	<pre>{   "accessToken": "string",   "refreshToken": {     "username": "string",     "token": "string",     "expireAt": "date time"   } }</pre>
401 Schema	<pre>{   "errorCode": number,   "errorMessage": "string" }</pre>

### Logout API

پس از اتمام تعامل با درگاه می‌توان با فراخوانی Logout API و ارسال `refreshToken` دریافتی در هنگام `login`، به درگاه اعلام نمود که JWT مرتبط را معدوم نماید. مشخصات `request` ارسالی به این API و `response` دریافتی به شرح جدول زیر است:

Logout API	
Request	
Method	post
Header	Authorization: Bearer accessToken
Body Content-Type	application/json
Parameters	refreshToken: "string"
Response	
Status Code	204 NoContent

## سرویس توکن احراز هویت (Token Service)

### Refresh

با استفاده از این API توکن احراز هویت JWT فعلی نوسازی شده و یک JWT جدید ارائه می‌شود. پارامترهای ارسالی به این API در کلاس [JwtRefreshRequest](#) توصیف شده‌اند. پاسخ دریافتی نیز مطابق کلاس [AuthResponse](#) است. مشخصات request ارسالی به این API و response دریافتی به شرح جدول زیر است:

Refresh API	
Request	
Method	post
Header	Authorization: Bearer accessToken
Body Content-Type	application/json
Schema	<a href="#">JwtRefreshRequest</a>
	{ "accessToken": "string", "refreshToken": "string" }
Response	
Status Code	200 Success / 400 BadRequest
200 Schema	<a href="#">AuthResponse</a>
	{ "accessToken": "string", "refreshToken": { "username": "string", "token": "string", "expireAt": "date time" } }
400 Schema	{ "errorCode": number, "errorMessage": "string" }

## Revoke

عملکرد و ساختار این API همانند Logout API بوده و با فراخوانی آن JWT مرتبط ابطال خواهد شد. مشخصات request ارسالی به این API و response دریافتی به شرح جدول زیر است:

Revoke API	
Request	
Method	post
Header	Authorization: Bearer accessToken
Body Content-Type	application/json
Parameters	refreshToken: "string"
Response	
Status Code	204 NoContent

در نمونه برنامه پیوست این سند به نام ISG.Demo، فراخوانی‌های مربوط به احراز هویت و نوسازی JWT پیاده‌سازی شده و قابل بررسی برای توسعه‌دهندگان می‌باشد.

## سرویس امضا (Signing Service)

### ChangePassword

تغییر گذرواژه از دو طریق امکان‌پذیر است. توسط مدیر سیستم در بخش مدیریت کاربران گذرواژه تغییر داده شود یا با فراخوانی ChangePassword API این کار صورت پذیرد. پارامترهای ارسالی به این API در کلاس [ChangePasswordRequest](#) توصیف شده‌اند. مشخصات request ارسالی به این API و response دریافتی به شرح جدول زیر است:

ChangePassword API	
Request	
Method	Post
Header	Authorization: Bearer accessToken
Body Content-Type	application/json
Schema	<a href="#">ChangePasswordRequest</a>
	{ "userName": "string", "password": "string" }
Response	
Status Code	200 Success
200 Schema	Boolean

### AllTemplates

این API فهرست الگوهای تعریف شده در حساب کاربری را ارائه می‌دهد. جهت بررسی اقلام الگو و کاربرد آن به بخش [Template](#) مراجعه نمایید. در فراخوانی این API نیازی به ارسال پارامتر وجود نداشته و داشتن JWT معتبر کفایت می‌نماید. مشخصات request ارسالی به این API و response دریافتی به شرح جدول زیر است:

AllTemplates API	
Request	
Method	Post
Header	Authorization: Bearer accessToken
Body Content-Type	application/json
Schema	No Parameters
Response	
Status Code	200 Success / 400 BadRequest
200 Schema	<a href="#">Template[]</a>
	<pre>[   {     "name": "string",     "direction": "string",     "contentType": number,     "signType": number,     "callbackUrl": "string",     "callbackType": number,     "hint": "string",     "title": "string",     "letSignerDownload": boolean,     "validityDuration": number,     "handwriteSignature": number,     "issuersDn": "string",     "version": "string" *   } ]</pre>
400 Schema	<pre>{   "errorCode": number,   "errorMessage": "string" }</pre>

\* در نسخه جاری مقدار version برای الگوها همیشه ۱.۰ است.

## TemplateInfo

این API با دریافت نام الگو مشخصات الگوی موردنظر را ارائه می‌دهد. پارامترهای ارسالی به این API در کلاس [TemplateRequest](#) توصیف شده‌اند. پاسخ دریافتی نیز مطابق کلاس [Template](#) است. مشخصات request ارسالی به این API و response دریافتی به شرح جدول زیر است:

TemplateInfo API	
Request	
Method	Post
Header	Authorization: Bearer accessToken
Body Content-Type	application/json
Schema	<a href="#">TemplateRequest</a>
	<pre>{   "templateName": "string" }</pre>

		}
Response		
Status Code		200 Success / 400 BadRequest
200 Schema		<a href="#">Template</a>
		{ "\"name\": \"string\", "\"direction\": \"string\", "\"contentType\": number, "\"signType\": number, "\"callbackUrl\": \"string\", "\"callbackType\": number, "\"hint\": \"string\", "\"title\": \"string\", "\"letSignerDownload\": boolean, "\"validityDuration\": number, "\"handwriteSignature\": number, "\"issuersDn\": \"string\", "\"version\": \"string\" * }
400 Schema		{ "\"errorCode\": number, "\"errorMessage\": \"string\" }

## TemplateStructure

در صورتی که محتوای ارسالی برای امضا از نوع json باشد، لازم است ساختار نمایش اطلاعات آن به صورت html در الگوی مرتبط تعریف شود تا درگاه، اطلاعات json را مطابق با ساختار تعریف شده، نمایش دهد. API TemplateStructure ساختار الگوی json را ارائه می‌دهد.

**نکته:** ساختار نمایش اطلاعات که به صورت html تعریف می‌شود با استفاده از قابلیت data-binding موجود در HTML5 عمل خواهد نمود. به دلایل حقوقی-امنیتی باید المنت‌های html موجود در ساختار ارایه شده، نظیر به نظیر با المنت‌های اطلاعات ارسالی در محتوای json یکسان باشند تا امضاکننده، تمامی اطلاعات را پیش از امضا مشاهده نماید. در صورت سازگار نبودن ساختار نمایش html با اطلاعات json و عدم نمایش حتی یک المنت از محتوای ارسالی، درگاه امضا، محتوا را معتبر ندانسته، آن را نمایش نداده و اجازه امضای محتوا را نخواهد داد.

پارامترهای ارسالی به این API در کلاس [TemplateRequest](#) توصیف شده‌اند. پاسخ دریافتی نیز مطابق کلاس [TemplateBlob](#) است. مشخصات request ارسالی به این API و response دریافتی به شرح جدول زیر است:

TemplateStructure API	
Request	
Method	Post
Header	Authorization: Bearer accessToken

Body Content-Type	application/json
Schema	<a href="#">TemplateRequest</a>
	{ "templateName": "string" }
Response	
Status Code	200 Success / 400 BadRequest
200 Schema	<a href="#">TemplateBlob</a>
	{ "structure": "html string", "logo": { "name": "string", "mediaType": "string", "content": "base64 string" } }
400 Schema	{ "errorCode": number, "errorMessage": "string" }

## GetSigningToken

با استفاده از این API محتوایی که باید امضا شود برای درگاه ارسال شده و در مقابل، یک شناسه ارجاع به نام SigningToken دریافت می‌گردد. هنگام فراخوانی این API اگر نام الگو ارسال شود، تمامی پارامترهای لازم به صورت خودکار از الگو برداشت خواهد شد و فقط ارسال اطلاعات محتوا به همراه نام الگو الزامی خواهد بود. در صورتی که هر پارامتر دیگری در این API مقداردهی و ارسال شود، اولویت با پارامتر ارسالی بوده و از مقدار مشابه آن در الگو صرف‌نظر خواهد شد.

برای محتوای نوع json ارسال نام الگو و وجود ساختار html جهت نمایش اطلاعات در آن الگو الزامی است. برای سایر انواع محتوا، ارسال نام الگو اختیاری می‌باشد. الگوی دارای ساختار، می‌تواند آرم سازمانی متفاوتی از آنچه در حساب کاربری معرفی شده است داشته باشد، هنگام ایجاد یک ساختار جدید، به صورت پیش‌فرض آرم سازمانی در ساختار الگو قرار می‌گیرد. در صورت تمایل می‌توانید برای الگوهای مربوط به json آرم سازمانی جداگانه‌ای معرفی نمایید.

هنگام ارسال محتوا به این نکته توجه داشته باشید که انواع مختلف data در پروتکل html رفتار متفاوتی دارند. به عنوان مثال کاراکترهای / و + که در بعضی از base64 string ها وجود دارند، در URL جزو کاراکترهای غیرمجاز و دارای ریسک شناسایی می‌شوند. یا ارسال اطلاعات html به دلیل کاراکترهای < و > ... به صورت عادی پذیرفته نمی‌شود.

در هنگام ارسال محتوا به نکات زیر توجه داشته باشید:

با توجه به اینکه هم امکان امضای رمزگذاری شده هست، هم مدل امضای Plain، اگر از مدل امضای رمزگذاری شده استفاده می‌گردد، باید محتوای ارسالی برای امضا، با توجه به توضیحاتی که برای [EncryptData](#) آورده شده است، یک رشته Base64 رمزگذاری شده باشد و همینطور مقدار پارامتر IsEncrypted، در این Api true ارسال گردد.

اما اگر از مدل امضای Plain استفاده می‌شود، مقدار پارامتر IsEncrypted باید false ارسال شود و توجه به نکات ذیل نیز حائز اهمیت است:

- ✚ اطلاعات فایل pdf را ابتدا به base64 string تبدیل نموده سپس URL Encode نمایید.
  - ✚ اطلاعات json را ابتدا اطمینان حاصل کنید که UTF-8 است، سپس URL Encode نموده و ارسال نمایید.
  - ✚ اطلاعات نوع html را به صورت Html Encode ارسال کنید.
  - ✚ اطلاعات text را به صورت URL Encode ارسال نمایید.
  - ✚ اطلاعات Chakad را به صورت text که URL Encode شده، ارسال نمایید.
- متناسب با نوع محتوای ارسالی، محدودیتی در نوع امضا وجود دارد:
- ✚ محتوای از نوع Text و Html را می توان با انواع مختلف امضا، امضا نمود.
  - ✚ محتوای از نوع pdf فقط قابلیت امضای pdf دارد.
  - ✚ محتوای از نوع json را باید (detached) CMS امضا کرد.
  - ✚ محتوای از نوع Chakad فقط قابلیت امضای CMS(attach) دارد.
- توجه به این نکته الزامی است که محتوا از نوع چکاد، با utf8 encoding امضا می شود و صرفاً برای چک بانک مرکزی، کاربرد دارد.

تمامی این موارد را می توانید در برنامه نمونه‌ای به نام ISG.Demo که در پیوست این سند قرار دارد ملاحظه فرمایید. در صورتی که محتوای ارسالی از نوع text یا html بوده و امضای درخواستی از نوع pdf باشد، درگاه به صورت خودکار محتوا را به pdf نسخه ۱.۶ (acrobat 7) تبدیل می نماید. اما اگر محتوای ارسالی از نوع pdf باشد، درگاه می تواند نسخه‌های پایین تر از ۱.۶ را به نسخه ۱.۶ تبدیل نموده و بقیه نسخه‌ها را بدون تغییر ارایه دهد. تبدیل نسخه pdf پایتتر از ۱.۶ به نسخه ۱.۶ به صورت پیش فرض غیرفعال بوده و توسط مدیر وب سرور درگاه قابل فعال شدن است. بنابراین توصیه می گردد اگر محتوای ارسالی به درگاه امضا، از نوع pdf می باشد، نسخه آن ۱.۶ به بالا باشد.

نسخه ۱.۶ کمترین شماره نسخه برای برخی از فعالیت‌های خاص بر روی سند، از جمله افزودن مهر سازمانی است.

**نکته:** این گونه فعالیت‌های خاص، خارج از درگاه امضا انجام می شود و به عهده سیستم مربوطه است. درگاه امضا فقط به منظور ایجاد تسهیلات برای فعالیت‌های احتمالی آتی، این قابلیت را فراهم نموده است.

**نکته:** حجم محتوای ارسال شده برای امضا، با توجه به مقداری که توسط مدیر وب سرور تنظیم شده، قابل کنترل است. در GetSigningToken API به صورت اختیاری می توان با تعیین مشخصات امضاکننده و نوع پیام‌رسانی، همزمان با ارسال محتوا، پیامی هم برای گوشی همراه امضاکننده ارسال نمود تا جهت رویت و امضای محتوا به درگاه مراجعه نماید. در صورتی که تمایل به ارسال پیام اطلاع‌رسانی به گوشی همراه امضاکننده را ندارید، می توانید Notification -> Type را معادل QueueOnly قرار دهید تا بدون ارسال پیام، صف امضا تشکیل شود. برای ارجاع آن دسته از کاربران که فاقد نرم افزار لازم روی تلفن همراه خود هستند یا از طریق مرورگر قصد امضای محتوا را دارند، پس از دریافت SigningToken می توان مرورگر کاربر را با فرمت زیر به درگاه امضا هدایت نمود:

<Protocol>://<ISG URL>/SigningToken

مثال: <https://isg.pki.co.ir/1DF6A769-4E82-49AB-8835-3CB146E4F00>

همچنین با تعیین مشخصات گواهینامه امضاکننده، امکان امضای محتوا، محدود به دارنده آن گواهینامه خواهد بود. بدین منظور در پارامتر issuersDn می‌توان یک یا فهرستی از اسامی صادرکنندگان مجاز گواهینامه را قرار داد که در این روش، امکان امضای محتوا، فقط به دارندگان گواهینامه از این مراکز، محدود می‌گردد.

برای محدود کردن امضاکننده، روش دیگری هم وجود دارد که می‌توان در پارامتر signerSubjectDn، کدملی یک شخص را به صورت "SERIALNUMBER=NationalID" قرار داد که در این روش، امکان امضا به یک امضاکننده، محدود می‌گردد. ارسال پارامتر metadata به منظور دریافت آن در GetData API از دیگر اقلام اختیاری در فراخوانی این API است. این پارامتر در API GetSigningToken توسط بهره‌بردار ارسال و بدون هیچ تغییری در GetData API دریافت می‌شود. از این پارامتر به عنوان برقراری mapping اطلاعات درگاه امضا با برنامه کاربردی بهره‌بردار استفاده می‌گردد. بدین صورت که بهره‌بردار در زمان ارسال محتوا به درگاه امضا، در API GetSigningToken در این پارامتر، مقدار کلیدی خود را قرار می‌دهد و پس از دریافت محتوای امضا در API GetData می‌تواند با دریافت آن، ارتباط برنامه کاربردی خود با محتوای امضای دریافت‌شده را برقرار نماید.

هرگاه امضاکننده اقدام به امضای محتوا نماید، در صورتی که امضا و گواهینامه امضاکننده معتبر باشند، درگاه امضا شدن محتوا را به نشانی callback اعلام می‌نماید.

در پارامتر [callbackType](#) نوع ارسال نتیجه‌ی امضا به بهره‌بردار مشخص می‌گردد. دو نوع callback وجود دارد: client-side و Server-side.

پارامتر callbackurl نیز نشانی فراخواننده سرویس می‌باشد و با توجه به نوع callback مقداردهی می‌گردد. بدین صورت که اگر نوع callback به صورت client-side باشد، بهره‌بردار باید متدی پیاده‌سازی کرده باشد که از جنس HttpGet بوده و پارامتر callbackurl را نیز به شکل ذیل مقداردهی نماید:

```
<Protocol>://<Beneficiary URL>/getResult  
https://isg.pki.co.ir/demo/home/getresult
```

و اگر نوع callback از نوع server-side باشد، بهره‌بردار باید متدی از جنس HttpPost پیاده‌سازی کرده باشد و مقدار پارامتر callbackurl نیز به شکل ذیل مقداردهی شود:

```
<Protocol>://<Beneficiary URL>/postResult  
https://isg.pki.co.ir/demo/home/postresult
```

**نکته:** در صورتی که نشانی callback از نوع server-side باشد، درگاه یک http request به نشانی callback ارسال نموده و در انتظار پاسخ http status باقی می‌ماند تا نتیجه را به امضاکننده اعلام نماید. توضیحات بیشتر در GetData API آورده شده است.

**نکته:** با توجه به اینکه اطلاعات مربوط به پیام‌رسانی به گوشی همراه (Push Notification)، تعیین امضاکننده خاص (Signer Subject DN) و metadata در الگوها تعریف نمی‌شوند، در صورت نیاز، باید در هر درخواست ارسال شوند. پارامترهای ارسالی به این API در کلاس [SigningRequest](#) توصیف شده‌اند. پاسخ دریافتی نیز مطابق کلاس [SigningResponse](#) است. مشخصات request ارسالی به این API و response دریافتی به شرح جدول زیر است:

GetSigningToken API	
Request	



Method	post
Header	Authorization: Bearer accessToken
Body Content-Type	application/json
Schema	<a href="#">SigningRequest</a>
	<pre>{   "dataToSign": "string",   "templateName": "string",   "direction": "string",   "contentType": number,   "signType": number,   "callbackType": number,   "callbackUrl": "string",   "hint": "string",   "title": "string",   "letSignerDownload": boolean,   "validityDuration": number,   "issuersDn": [     "string"   ],   "signerSubjectDn": "string",   "handwriteSignature": number,   "contentImage": {     "image": "string",     "imageType": "string",     "page": number,     "lowerLeftX": number,     "lowerLeftY": number,     "upperRightX": number,     "upperRightY": number   },   "metadata": "string",   "notification": {     "token": "string",     "type": number   },   "IsEncrypted" : boolean }</pre>
Response	
Status Code	200 Success / 400 BadRequest
200 Schema	<a href="#">SigningResponse</a>
	<pre>{   "signingToken": "string",   "notificationOutput": {     "notifSucceedCount": number,     "notificationResponseList":       IEnumerable&lt;<a href="#">notificationResponse</a>&gt;,     "issuccess": Boolean,   } }</pre>

	<pre>"description": "string", "errorCode": number, "errorMessage": "string", }, "succeeded": boolean, "errorCode": number, "errorMessage": "string" }</pre>
400 Schema	<pre>{ "errorCode": number, "errorMessage": "string" }</pre>

## CheckStatus

پس از ارسال محتوا و دریافت شناسه ارجاع (Signing Token) هر زمان که لازم باشد می‌توان با فراخوانی این API وضعیت جاری محتوا را بررسی نمود. پارامترهای ارسالی به این API در کلاس [GeneralRequest](#) توصیف شده‌اند. پاسخ دریافتی نیز مطابق [ContentStatus](#) است. مشخصات request ارسالی به این API و response دریافتی به شرح جدول زیر است:

CheckStatus API	
Request	
Method	Post
Header	Authorization: Bearer accessToken
Body Content-Type	application/json
Schema	<a href="#">GeneralRequest</a>
	<pre>{ "signingToken": "string" }</pre>
Response	
Status Code	200 Success / 400 BadRequest
200 Schema	" <a href="#">contentStatus</a> ": number
400 Schema	<pre>{ "errorCode": number, "errorMessage": "string" }</pre>

## GetData

هرگاه امضاکننده اقدام به امضای محتوا نماید، در صورتی که امضا و گواهینامه امضاکننده معتبر باشند، درگاه امضا شدن محتوا را به نشانی callback اعلام می‌نماید. در این زمان با فراخوانی این API تمامی اطلاعات مرتبط با امضا ارائه خواهد شد.

**نکته:** در صورتی که نشانی callback از نوع server-side باشد، درگاه یک http request به نشانی callback ارسال نموده و در انتظار پاسخ http status باقی می‌ماند تا نتیجه را به امضاکننده اعلام نماید. بنابراین توصیه می‌شود بهره‌برداران پس از دریافت اعلام امضا شدن محتوا، ابتدا پاسخ 200 http status را به درگاه بازگردانده، سپس با یک وقفه کوتاه API GetData را فراخوانی نمایند تا سرور درگاه با کارایی مناسب وظایف خود را انجام دهد. ایجاد یک queue با وقفه‌ای در حدود ۵۰۰ میلی ثانیه فرصت لازم

را در اختیار سرور درگاه قرار می‌دهد تا همزمانی امضاکنندگان را مدیریت نموده و پایان فرآیند را به امضاکننده اعلام نماید. این نکته نیز در ISG.Demo در QueueService پیاده‌سازی شده و قابل بررسی می‌باشد. پارامترهای ارسالی به این API در کلاس [GeneralRequest](#) توصیف شده‌اند. پاسخ دریافتی نیز مطابق کلاس [SignatureResponse](#) است. مشخصات request ارسالی به این API و response دریافتی به شرح جدول زیر است:

GetData API	
Request	
Method	post
Header	Authorization: Bearer accessToken
Body Content-Type	application/json
Schema	<a href="#">GeneralRequest</a>
	{ "signingToken": "string" }
Response	
Status Code	200 Success / 400 BadRequest
200 Schema	<a href="#">SignatureResponse</a>
	{ "certificate": "base64 string", "serialNumber": "string", * "digest": "string", ** "signature": "base64 string", "metadata": "string", "signType": number, "sendTime": "string", "senderIP": "string", "signTime": "string", "signerIP": "string", "issuerDn": "string", "subjectDn": "string", "signerCn": "string", *** "thumbprint": "string" }
400 Schema	{ "errorCode": number, "errorMessage": "string" }

\* مقدار serialNumber شماره سریال گواهینامه امضاکننده بر اساس CA صادر کننده گواهینامه است.

\*\* در صورتی که امضا از نوع PDFSign باشد Digest امضا شده دریافت می‌شود. در سایر انواع امضا، Digest امضا نشده دریافت می‌گردد.

\*\*\* مقدار signerCn اشاره به common name گواهینامه امضاکننده دارد. مقدار این آیتم در گواهینامه‌های صادره از سوی مراکز میانی صدور گواهینامه‌های کشوری متفاوت بوده اما در هر صورت مشخص‌کننده شناسه یکتای صاحب گواهینامه می‌باشد. به عنوان مثال در گواهینامه‌های کارت ملی، شناسه ملی هر فرد در آیتم common name گواهینامه ذخیره شده است.

**نکته:** وجود کاراکترهای / و + در بعضی از base64 string ها موجب می شود تا مقادیر base64string از قبیل certificate و signature به صورت خودکار URL Encode شوند. بنابراین هنگام دریافت مقادیر base64string با استفاده از متد IsBase64String که در مولفه DTO فراهم شده است، مقدار دریافتی را بررسی نمایید. در صورتی که IsBase64String مقدار false برمی گرداند، لازم است base64string دریافتی را URL Decode نموده، سپس در توابعی مانند Verify استفاده کنید. در استفاده از توابع PKTB لازم است رشته های ارسالی از نوع Unicode (UTF-16) باشند. هم در PKTB توابع لازم برای تبدیل رشته های متنی به Unicode وجود دارد و هم می توان با داشتن متن اصلی (message)، آرایه بیتی Unicode آن را استخراج و در توابعی نظیر Verify استفاده نمود. اطلاعات html و xml نیز به دلیل وجود کاراکترهای < و > ... به صورت عادی پذیرفته نمی شود و باید تمهیدات لازم در نقل و انتقال و دریافت این گونه اطلاعات در نظر گرفته شود. جهت بررسی دقیق موضوع به کلاس SignatureExtensions در پروژه ISG.Demo مراجعه نمایید.

## RemoveData

با فراخوانی این API و ارسال شناسه ارجاع (Signing Token)، تمامی اطلاعات مربوط به محتوا و امضای آن حذف خواهند شد. متناسب با پارامترهای ارسالی در SigningRequest مقدار بازگشتی این API عددی بین ۳ تا ۵ خواهد بود که بیانگر حذف موفقیت آمیز تعداد رکوردهای مرتبط با محتوای ارسالی و امضای آن است. پارامترهای ارسالی به این API در کلاس [GeneralRequest](#) توصیف شده اند. مشخصات request ارسالی به این API و response دریافتی به شرح جدول زیر است:

RemoveData API	
Request	
Method	delete
Header	Authorization: Bearer accessToken
Body Content-Type	application/json
Schema	<a href="#">GeneralRequest</a>
	{ "signingToken": "string" }
Response	
Status Code	200 Success / 400 BadRequest
200 Schema	RowsAffected": number
400 Schema	{

```

"errorCode": number,
"errorMessage": "string"
}

```

## ResendNotification

در صورتی که پیام اطلاع‌رسانی با موفقیت ارسال شده ولی به گوشی همراه کاربر نرسیده باشد، با استفاده از این API می‌توان اقدام به ارسال مجدد همان پیام قبلی نمود. پیش‌نیاز استفاده از این API ارسال اطلاعات پیام‌رسانی در هنگام ارسال محتوا با استفاده از `GetSigningToken` API است. این API فقط برای محتوایی اقدام به ارسال مجدد پیام اطلاع‌رسانی می‌کند که آن محتوا هنوز امضا نشده و منقضی هم نشده باشد. در سایر حالت‌ها، کد خطای ۲۴ (`ContentExpired`) را دریافت می‌نمایید. پارامترهای ارسالی به این API در کلاس `GeneralRequest` توصیف شده‌اند. پاسخ دریافتی نیز مطابق کلاس `NotificationOutput` است. لازم به توضیح است که پارامتر `notifSucceedCount` تعداد نوتیفیکیشن ارسال شده‌ی موفق به کاربر (بر حسب تعداد برنامه‌های امضای کاربر) را نمایش می‌دهد. پارامتر `notificationResponseList` فهرست نوتیفیکیشن‌های ارسال شده را نمایش می‌دهد که در این فهرست، علاوه بر ارسال‌های موفق، ارسال‌های ناموفق هم قرار دارد.

مشخصات request ارسالی به این API و response دریافتی به شرح جدول زیر است:

ResendNotification API	
Request	
Method	post
Header	Authorization: Bearer accessToken
Body Content-Type	application/json
Schema	<code>GeneralRequest</code>
	{ "signingToken": "string" }
Response	
Status Code	200 Success / 400 BadRequest
200 Schema	<code>NotificationOutput</code>
	{ "notifSucceedCount": number, "notificationResponseList": IEnumerable< <code>notificationResponse</code> >, }
400 Schema	{ "errorCode": number, "errorMessage": "string" }

## سرویس Notification (Notification Service)

این سرویس دارای دو API است:

### GetSigningQueueList

این API برای نمایش فهرست درخواست‌های امضای ارسال شده به کاربر می‌باشد. به این صورت که بهره‌بردار، شناسه ملی شخص امضاکننده را ارسال می‌کند و فهرست درخواست‌های آن شخص را دریافت می‌نماید.

مشخصات request ارسالی به این API و response دریافتی به شرح جدول زیر است:

GetSigningQueueList API	
Request	
Method	post
Header	Authorization: Bearer accessToken
Body Content-Type	application/json
Schema	<a href="#">SigningQueueRequest</a>
	<pre>{   "nationalCode": "string" }</pre>
Response	
Status Code	200 Success / 400 BadRequest
200 Schema	<a href="#">SigningQueueResponse</a>
	<pre>{   "notificationQueueList":   IEnumerable&lt;NotificationQueue&gt;, }</pre>
400 Schema	<pre>{   "errorCode": number,   "errorMessage": "string" }</pre>

### RemoveFromQueue

این API برای نمایش حذف درخواست امضای ارسال شده به کاربر می‌باشد.

مشخصات request ارسالی به این API و response دریافتی به شرح جدول زیر است:

RemoveFromQueue API	
Request	
Method	post
Header	Authorization: Bearer accessToken
Body Content-Type	application/json

	Schema	<a href="#">GeneralRequest</a>
		{ "signingToken": "string" }
Response		
	Status Code	200 Success / 400 BadRequest
	200 Schema	<a href="#">GeneralResponse</a>
		{ "issuccess": Boolean }
	400 Schema	{ "errorCode": number, "errorMessage": "string" }

## سرویس Tools (Tools Service)

این سرویس دارای یک API است:

### EncryptData

این API اقدام به Encrypt کردن محتوای امضا می‌نماید. به این صورت که بهره‌بردار باید محتوایی که می‌خواهد برای درگاه امضا ارسال کند، به همراه یک رمز که باید برای امضاکننده ایجاد و ارسال شود را دریافت و این اطلاعات را رمزگذاری کرده و محتوای رمزگذاری شده را به صورت یک رشته base64 برمی‌گرداند.

در هنگام ارسال محتوا به نکات زیر توجه داشته باشید:

اطلاعات فایل pdf را ابتدا به base64 string تبدیل نموده سپس URL Encode نمایید.

اطلاعات json را ابتدا اطمینان حاصل کنید که UTF-8 است، سپس URL Encode نموده و ارسال نمایید.

اطلاعات نوع html را به صورت HTML Encode ارسال کنید.

اطلاعات text را به صورت URL Encode ارسال نمایید.

Encrypt API	
Request	
Method	4
Header	Authorization: Bearer accessToken
Body Content-Type	application/json
Schema	<a href="#">GeneralRequest</a>
	<pre>{   "data": "string",   "Pin": "string" }</pre>
Response	
Status Code	200 Success / 400 BadRequest
200 Schema	<pre>{"EncryptedData": "string",   "IsEncrypted": "bool" }</pre>



## 400 Schema

```
{  
  "errorCode": number,  
  "errorMessage": "string"  
}
```

## شمای تبادل اطلاعات (Schemas)

در این بخش schema های مورد استفاده در API های سرویس ها توصیف شده اند. تمامی schema های استفاده شده در مؤلفه ISG.DTO وجود داشته و توسعه دهندگان .NET و Java می توانند از نسخه سازگار با پلتفرم خود استفاده نمایند.

AuthRequest		
UserName	String	Not null
Password	String	Not null

- در نسخه آتی مکانیزم احراز هویت تغییر خواهد نمود و به جای ارسال نام کاربری و گذرواژه یک challenge از سرویس مربوط دریافت و Sign آن به منظور احراز هویت ارسال خواهد شد.

AuthResponse		
AccessToken	String	Not null
RefreshToken	<a href="#">JwtRefreshToken</a>	Not null

- طول عمر access-token و refresh-token متناسب با نیازمندی بهره بردار قابل تنظیم می باشد.

CertificateValidationRequest		
Certificate	Base64 string	Not null

ChangePasswordRequest		
UserName	String	Not null
CurrentPassword	String	Not null
NewPassword	String	Not null

ContentImage		
Image	Base64 String	Not null
ImageType	<a href="#">ImageType</a> (String)	Not null
Page	Number (Int 1 to 32767)	Not null
LowerLeftX	Number (float)	Not null
LowerLeftY	Number (float)	Not null
UpperRightX	Number (float)	Not null
UpperRightY	Number (float)	Not null

- ContentImage فقط برای ارسال تصویر برای محتوای نوع PDF کاربرد دارد. این تصویر پیش از امضا نمایش داده نمی شود ولی در صورتی که امضاکننده اجازه دانلود PDF امضا شده را داشته باشد و پس از امضا آن را دانلود نماید، تصویر ارسالی به عنوان تگ امضا در موقعیت مشخص شده قرار خواهد داشت. همچنین با فراخوانی API GetData سند PDF دریافتی شامل تصویر ارسالی نیز خواهد بود.
- مقدار Image می بایست Base64String و URL Encode شده باشد.
- موقعیت تصویر بر اساس +X و +Y در محور مختصات محاسبه می گردد. به بیان دیگر نقطه (0, 0) در سمت چپ و پایین صفحه PDF قرار دارد.

ErrorResult		
ErrorCode	Int (4)	Default
ErrorMessage	String	Allow null

- فهرست کدهای خطا و شرح آن در جدول [Error Codes](#) در انتهای بخش ضمیمه آورده شده است.

GeneralRequest		
SigningToken	String	Not null

GeneralResponse		
IsSuccess	boolean	Not null
Description	String	Not null
ErrorCode	int	Not null
ErrorMessage	String	Not null

JwtRefreshRequest		
AccessToken	String	Not null
RefreshToken	Base64 String	Not null

JwtRefreshToken		
UserName	String	Not null
Token	Base64 String	Not null
ExpireAt	DateTime	Not null

JwtToken		
AccessToken	String	Not null
RefreshToken	<a href="#">JwtRefreshToken</a>	Not null

Logo		
Name	String	Not null
MediaType	String	Not null
Content	Base64 String	Not null

Notification		
Token	String	Not null (App Token, National ID, Mobile Number, Email Address)
Type	<a href="#">NotificationType</a>	Not null

- در صورتی که مقدار App Token نرم افزار گوشی همراه (mkeyone) را در اختیار داشته باشید، می توانید با مشخص نمودن Type (Apple Token or Google Token) توکن مذکور را جهت مشخص نمودن تلفن همراه گیرنده پیام، ارسال نمایید.

در غیر این صورت با ارسال یکی از پارامترهای کد ملی، شماره تلفن همراه یا نشانی پست الکترونیکی که موقع ثبت نام در mkeyone توسط کاربر ارائه شده است، تلفن همراه گیرنده پیام به صورت خودکار توسط API مربوط شناسایی خواهد شد. در این شرایط باید مقدار Type را برابر با Local قرار دهید.

NotificationRequest		
Notification	<a href="#">Notification</a>	Not null
SigningToken	String	Not null
Owner	String	Not null
Subject	String	Not null

- به صورت پیش فرض نام فارسی فراخواننده API به عنوان Owner و عنوان معرفی شده در پارامتر Title (در هنگام ارسال محتوا توسط GetSigningToken API) به عنوان Subject در نظر گرفته شده و در پیام دعوت به امضا در تلفن همراه، به کاربر نمایش داده می شوند.

NotificationResponse		
Result	String	Not null
NotificationStatus	<a href="#">NotificationStatus</a>	Not null
Success	Boolean	Not null

NotificationOutput		
NotifSucceedCount	Int	Not null
NotificationResponseList	<a href="#">NotificationResponse</a>	Not null
Issuccess	Boolean	Not null
Description	String	Not null
ErrorCode	Int	Not null
ErrorMessage	String	Not null

SignatureResponse		
Succeeded	Boolean	Not null
Certificate	Base64 String	Not null
SerialNumber	String	Not null
Digest	String	Not null
Signature	Base64 String	Not null
SignType	<a href="#">SignType</a>	Not null
Metadata	String	Allow null
SendTime	DateTime	Not null
SenderIP	String	Not null
SignTime	DateTime	Not null
SignerIP	String	Not null
IssuerDn	String	Not null
SubjectDn	String	Not null

SignerCn	String	Not null
Thumbprint	String	Not null

- مقدار Serial Number شماره سریالی است که از سوی CA صادر کننده به گواهینامه اختصاص داده می‌شود.
- در صورتی که امضا از نوع PDFSign باشد Digest امضا شده دریافت می‌شود. در سایر انواع امضا، Digest امضا نشده دریافت می‌گردد.
- مقدار SignerCn مقدار Common Name درج شده در SubjectDn گواهینامه امضاکننده است.

SigningRequest		
DataToSign	Base64 string/Encoded Html/String	Not null
TemplateName	String	Allow null
Direction *	String	Allow null
ContentType	<a href="#">ContentType</a>	Allow null
SignType	<a href="#">SignType</a>	Allow null
CallbackType	<a href="#">CallbackType</a>	Allow null
CallbackUrl	String	Allow null
Hint	String	Allow null
Title	String	Allow null
LetSignerDownload	Boolean	Allow null
ValidityDuration **	Number	Allow null (default value: 14400)
IssuersDn ***	String Array	Allow null
SignerSubjectDn ****	String	Allow null
HandwriteSignature *****	<a href="#">HandwriteSignature</a>	Allow null
ContentImage	<a href="#">ContentImage</a>	Allow null
Metadata	String	Allow null
Notification	<a href="#">Notification</a>	Allow null

نوع DataToSign وابسته به نوع محتوای ارسالی (Content Type) می‌باشد.

- برای PDF ابتدا باید Base64 string، سپس Url Encode ارسال شود.
- برای HTML باید Encoded Html string ارسال شود.
- برای Text ارسال Text با توجه به URL Encode شدن آن کفایت می‌نماید.
- برای Json ابتدا باید UTF-8، سپس Url Encode ارسال گردد.
- برای Chakad ارسال Text با توجه به URL Encode شدن آن کفایت می‌نماید.

\* پارامتر Direction فقط هنگام ارسال Text و Chakad کاربرد دارد.

\*\* از نسخه ۲.۴.۳ (ISG.Infrastructure) حداکثر زمان اعتبار محتوای امضا نشده، ۱۰ روز یا معادل ۱۴،۴۰۰ دقیقه (با توجه به مقدار تنظیم شده در فایل Setting) می‌باشد. مقادیر صفر و بزرگتر از ۱۴،۴۰۰ به صورت خودکار به ۱۴،۴۰۰ (با توجه به مقدار تنظیم شده در فایل Setting) تغییر می‌کنند.

\*\*\* با ارسال پارامتر IssuersDn به صورت آرایه‌ای از string، گواهینامه‌های مجاز به امضا محدود به گواهینامه‌هایی می‌شوند که توسط یکی از CA های میانی که در این پارامتر ذکر شده‌اند صادر گردیده باشند. این مقدار برابر با مقدار SubjectDn ذکر شده در گواهینامه CA یا همان مقدار فیلد Issuer در گواهینامه امضاکننده است.

\*\*\*\* با ارسال پارامتر SignerSubjectDn گواهینامه مجاز به امضا، محدود می‌شود به یک گواهینامه خاص که فیلد SubjectDn آن دارای مقداری برابر با این پارامتر است. به بیان دیگر، فقط یک امضاکننده مشخص، مجاز به امضای محتوا خواهد بود.

\*\*\*\* در صورتی که قصد ارسال تصویر امضای فرد امضاکننده را با استفاده از پارامتر ContentImage برای امضای نوع PDF داشته باشید، باید مقدار پارامتر HandwriteSignature را معادل Content قرار دهید. ارسال هر مقدار دیگر برای پارامتر HandwriteSignature موجب صرف‌نظر از مقدار ContentImage خواهد شد.

SigningResponse		
SigningToken	String	Not null
NotificationResponse	<a href="#">NotificationOutput</a>	Not null
Succeeded	Boolean	Not null

SigningResult		
SigningToken	String	Not null

SigningQueueRequest		
NationalCode	String	Not null

SigningQueueResponse		
NotificationQueueList	IEnumerable<NotificationQueue>	Not null
Issuccess	boolean	Not null
Description	String	Not null
ErrorCode	int	Not null
ErrorMessage	String	Not null

NotificationQueue		
SigningToken	String	Not null
Owner	String	Not null
Subject	String	Not null
Gateway	String	Not null
ImageURL	String	Not null
Date	DateTime	Not null

Template		
Name	String	Not null
Direction	String	Not null
ContentType	<a href="#">ContentType</a>	Not null
SignType	<a href="#">SignType</a>	Not null
CallbackUrl	String	Not null
CallbackType	<a href="#">CallbackType</a>	Not null
Hint	String	Allow null

Title	String	Not null
LetSignerDownload	Boolean	Not null
ValidityDuration	Int (2)	Min: 1, Max:14400
HandwriteSignature	<a href="#">HandwriteSignature</a>	Not null
IssuersDn	String	Allow null
Version	String	Always 1.0

ایجاد Template، تعامل با سرویس Signing و به طور مشخص یکی از پرکاربردترین API آن یعنی GetSigningToken را تسهیل می‌نماید. در صورتی که برای تیپ‌های مختلف امضا Template های مجزا تعریف شود، هنگام فراخوانی GetSigningToken ارسال پارامترهای DataToSign و TemplateName کفایت نموده و باقی پارامترها به صورت خودکار از Template معرفی شده مقداردهی خواهند شد.

TemplateBlob		
Structure	String	Allow null
Logo	<a href="#">Logo</a>	Not null

- به ازای هر Template این امکان وجود دارد که امضاکننده تصویر logo متفاوتی را مشاهده نماید.
- در صورتی که نوع محتوای ارسالی JSON باشد، لازم است ساختار نمایش اطلاعات JSON در قالب HTML تعریف و در آیتم Structure ذخیره گردد.

TemplateRequest		
TemplateName	String	Not null

CallbackType		
ClientSide	1	Client-side callback
ServerSide	2	Server-side callback

CertificateStatus		
Valid	0	The certificate is valid.
InvalidPeriod	1	The certificate has expired.
InvalidChain	2	The certificate chain is invalid.
InvalidIntegrity	3	Certificate integrity is invalid.
InvalidKeyUsage	4	The certificate key usage is invalid.
OCSPRevoked	5	The certificate has been revoked (OCSP).
OCSPValidationUnKnown	6	Certificate validity status is unknown (OCSP).
CRLRevoked	7	The certificate has been revoked (CRL).
CRLOrOCSPValidationError	8	Certificate validation encountered a problem. One of the CRL or OCSP servers is not available.
CRLValidationUnKnown	9	Certificate validity status is unknown (CRL).
CRLAndOCSPValidationUnknown	10	Certificate validation encountered a problem. Both CRL or OCSP servers is not available.

ContentStatus		
Unknown	0	Unknown
NotExist	1	Not exist
NotSigned	2	Not signed yet
Expired	3	Expired
Signed	4	Signed
Delivered	5	Delivered

ContentType		
Html	1	Hyper Text Markup Language
Pdf	2	Portable Document Format
Text	3	Plain Text
Json	4	Java Script Object Notation
Chakad	5	Plain Text

• محتوای نوع Html و Text توسط تمامی انواع امضا (PDF, CMS attach, CMS detach, ESF, RSA)، قابل امضا کردن هستند.

- محتوای نوع Pdf فقط قابلیت امضای PDF دارد.
- محتوای نوع Json فقط قابلیت امضای CMS به روش detach دارد.
- محتوای نوع Chakad فقط قابلیت امضای CMS به روش attach دارد.

Directions		
LTR	ltr	Left to right



RTL	rtl	Right to left
Unset	unset	Unset

HandwriteSignature		
None	0	No signature image
Content	1	The signature image is received from ContentImage
Optional	2	The signer can add the signature image
Required	3	The signer must add the signature image

ImageType	
APNG	image/apng
GIF	image/gif
JPEG	image/jpeg
JPG	image/jpg
PNG	image/png
TIF	image/tif
TIFF	image/tiff

NotificationStatus		
Ready	0	Ready to send
Sent	1	Sent
Success	2	Accepted by delivery service
Failed	3	Not accepted by delivery service

NotificationType		
Google	0	Google token
Apple	1	Apple token
Local	2	Mobile number, National ID, Email
QueueOnly	3	Create a signature queue without push-notification.

SignType		
Unknown	0	Unknown
PDFSign	1	PDF Sign PKCS#1
CMSSign	2	CMS Detached Sign PKCS#7
ESFSign	3	ESF Sign PKCS#1
RSASign	4	RSA Sign PKCS#1
CMSAttachedSign	5	CMS Attached Sign PKCS#7

Error Codes		
InvalidCallerIP	1	نشانی فراخواننده سرویس نامعتبر است

InvalidUsernameOrPassword	2	نام کاربری یا گذرواژه نامعتبر است
UserIsNotEnable	3	کاربر غیر فعال است
TokenTimeout	4	توکن ارسالی منقضی شده است
Exception	5	خطای سیستمی
TokenIsUsed	6	محتوای درخواستی پیش از این امضا شده است
TokenNotFound	7	توکن ارسالی پیدا نشد
InvalidRequestOutputType	8	نوع تعریف شده برای خروجی امضا نامعتبر است
DastineError	9	خطای دستینه
NotImplementedException	10	این قابلیت هنوز پیاده سازی نشده است
SignatureNotVerified	11	امضا معتبر نیست
InvalidArchiveTokenId	12	توکن درخواست شده نامعتبر است
ValidationFailed	13	اطلاعات توکن نامعتبر است
InvalidSignType	14	نوع امضا نامعتبر است
InvalidFileType	15	فایل‌های {+} پشتیبانی نمی‌شوند
InvalidPosition	16	موقعیت تصویر امضا نادرست است
NotSignedYet	17	محتوای مرتبط هنوز توسط کاربر امضا نشده است.
InvalidCallbackType	18	نوع فراخوانی نشانی callback معتبر نیست.
InvalidCallbackUrl	19	نشانی callback معتبر نیست.
NotificationTokenNotFound	20	توکن پیام رسانی پیدا نشد.
UserIsTampered	21	حساب کاربری دستکاری شده است.
CallbackServerNotAvailable	22	سرور دریافت کننده‌ی نتیجه امضا در دسترس نیست.
ValidityDurationOutOfRange	23	مدت اعتبار توکن بیشتر از ۱۰ روز (+۱۴۴ دقیقه) پذیرفته نمی‌شود. (این مدت، توسط مدیر وب‌سرور قابل تغییر است)
ContentExpired	24	مدت اعتبار محتوای قابل امضا منقضی شده است.
AndroidNotificationAccountNotExist	25	حساب پیام رسانی Android وجود ندارد.
IOSNotificationAccountNotExist	26	حساب پیام رسانی iOS وجود ندارد.
InvalidTemplateName	27	قالب نمایش تحت وب، با نام درخواستی وجود ندارد.
AccessToSignedContentIsNotAllowed	28	دسترسی به محتوای امضا شده مجاز نیست.
NullContent	29	محتوای ارسالی تهی است.
TemplateNotFound	30	قالب محتوا پیدا نشد.
NotificationAccountNotFound	31	حساب پیام رسانی پیدا نشد.
NotificationOwnerNotFound	32	نام نمایشی ارسال کننده پیام پیدا نشد.
NotificationTokenNull	33	توکن پیام رسانی مقدار ندارد.
SignerNotFound	34	مشخصات امضاکننده پیدا نشد.

SignatureNotFound	35	اطلاعات امضا پیدا نشد .
InvalidContentType	36	نوع محتوا معتبر نیست .
NotificationRequestNotFound	37	اطلاعات درخواست پیام رسانی پیدا نشد .
InvalidRequestedContent	38	محتوای درخواستی نامعتبر است .
ContentNotReceived	39	محتوایی جهت امضا ارسال نشده است .
JsonToPdfNotSupported	40	از تبدیل json به pdf پشتیبانی نمی‌شود .
ISGNotificationNotConfigured	41	درگاه امضا برای ارسال پیام بیکربندی نشده است .
InvalidOwner	42	توکن و محتوای مرتبط با آن متعلق به کاربر دیگری است .
RequiredTitle	43	عنوان محتوا الزامی است .
RequiredJsonTemplateStructure	44	برای محتوای نوع json ساختار الگو الزامی است .
InvalidHandwriteSignature	45	نوع تصویر امضا معتبر نیست .
InvalidMaxLenDataToSign	46	حجم محتوای ارسال شده برای امضا معتبر نیست .
InvalidDataToSign	47	محتوای ارسال شده برای امضا معتبر نیست .
InvalidOCSP	48	گواهینامه امضای شما از سوی سامانه OCSP نامعتبر اعلام شده است .
NoCertificate	49	گواهینامه امضا کننده وجود ندارد .
UnavailableDSSService	50	سرویس DSS در دسترس نیست .
InvalidKeyUsages	51	کاربرد گواهی نامعتبر است .
InvalidAuthenticationAttempt	70	تلاش برای احراز هویت نامعتبر است .
AuthenticationLastChance	71	احراز هویت نامعتبر است . فقط یک بار دیگر می‌توانید تلاش کنید .
AccountLockedOut	72	حساب کاربری قفل شده است . با مدیر سیستم تماس بگیرید .
NotAllowedToLogin	73	شما مجاز به ورود به سیستم نیستید .
ChangePasswordError	74	تغییر گذرواژه با خطا مواجه شد .
DisableUserError	75	غیر فعال کردن کاربر با خطا مواجه شد .
RequiredUserName	76	نام کاربری الزامی است .
RequiredDisplayName	77	نام نمایشی الزامی است .
RequiredValidIP	78	دست کم یک نشانی آی پی الزامی است .
RequiredPassword	79	گذرواژه الزامی است .
MatchConfirmPassword	80	گذرواژه و تایید گذرواژه یکسان نیستند .
UserRegistrationException	81	خطا در هنگام ایجاد حساب کاربری
UserNotFound	82	کاربر مورد نظر پیدا نشد .
AccessTokenExpired	83	توکن دسترسی منقضی شده است .
RefreshTokenExpired	84	توکن بازسازی دسترسی منقضی شده است .

InvalidAccessToken	85	توکن دسترسی معتبر نیست .
InvalidRefreshToken	86	توکن بازسازی معتبر نیست .
InvalidRequestedToken	87	توکن درخواستی معتبر نیست .
InvalidPayload	88	اطلاعات احراز هویت معتبر نیست .
AccessViolation	89	دسترسی مجاز نیست.
WrongPassword	90	رمز عبور معتبر نیست.
RequiredLogo	91	آرم سازمانی الزامی است.
NoSignTools	92	ابراز امضا انتخاب نشده است.
NoRecord	93	رکورد موردنظر یافت نشد
OperationFailed	94	عملیات انجام نشد.
NoExistTemplate	95	این الگو وجود ندارد.
UnhandledException	100	یک خطای مدیریت نشده رخ داده است. لطفا با مدیر سیستم تماس بگیرید .