Desko ID reader Gateway

Application name:

DeskoIdScanGateway.exe

Optional parameters:

Parameter	Description
-port=9000	Defines the port for the the Web api (default is port 9000)

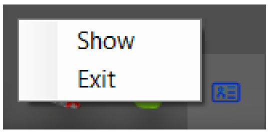
Usage:

Application must be started manually on Windows start or needs to be configured to start automatically with windows.

Close button of the main window is disabled on purpose to avoid accidental shutdown of the application which will result in the unavailability of the WebApi endpoint.

When started application will start minimized.

To access the main window, right click on the application icon in the Windows Tray area (next to the windows Clock). Right click on it and select **Show** menu item.



To close the application select **Exit** from the same menu as above

When device is not connected, the icon will change to a different one, with a red border to warn the user that the scan service si not available.

WebApi endpoints

1. **Desko**

Scan a document:

http://127.0.0.1:9000/api/scan

http://127.0.0.1:9000/api/scan?includeDocumentPicture=false

Accessing any of the 2 above endpoints using GET method will do a scan and will return a Json content with the detected fields and the Portrait ID from the ID document.

The document ID picture will NOT be returned with these calls as it has a big size. See below the endpoint that can be used to fetch the ID picture separately.

```
public class ScanResult
{
    public string ScanState { get; set; }

    public KeyValuePair<string, string>[] Fields { get; set; }

    public byte[] Portrait { get; set; } // base64encoded byte array with image data

    public byte[] IdPhoto { get; set; } // base64encoded byte array with image data

}
```

Check attached ScanResultSample.json for an example of the returned Json.

If includeDocumentPicture parameter is set to true, the response will contain also the ID picture, but keep in mind that this is usually big in size and the response will take longer to be returned.

http://127.0.0.1:9000/api/scan/documentPhoto

After a scan using one of the above methods, the id of the photo can be retrieved after that by using this endpoint. The response is a File content result of `"image/jpeg"` type (`base64encoded byte array with image data`).

http://127.0.0.1:9000/api/scan/status

This endpoint is used to get the status of the device connection and of the scan status. The response is a Json like the one below:

```
{"deviceState":"Plugged","scanState":” Idle” }
```

Device states values are:

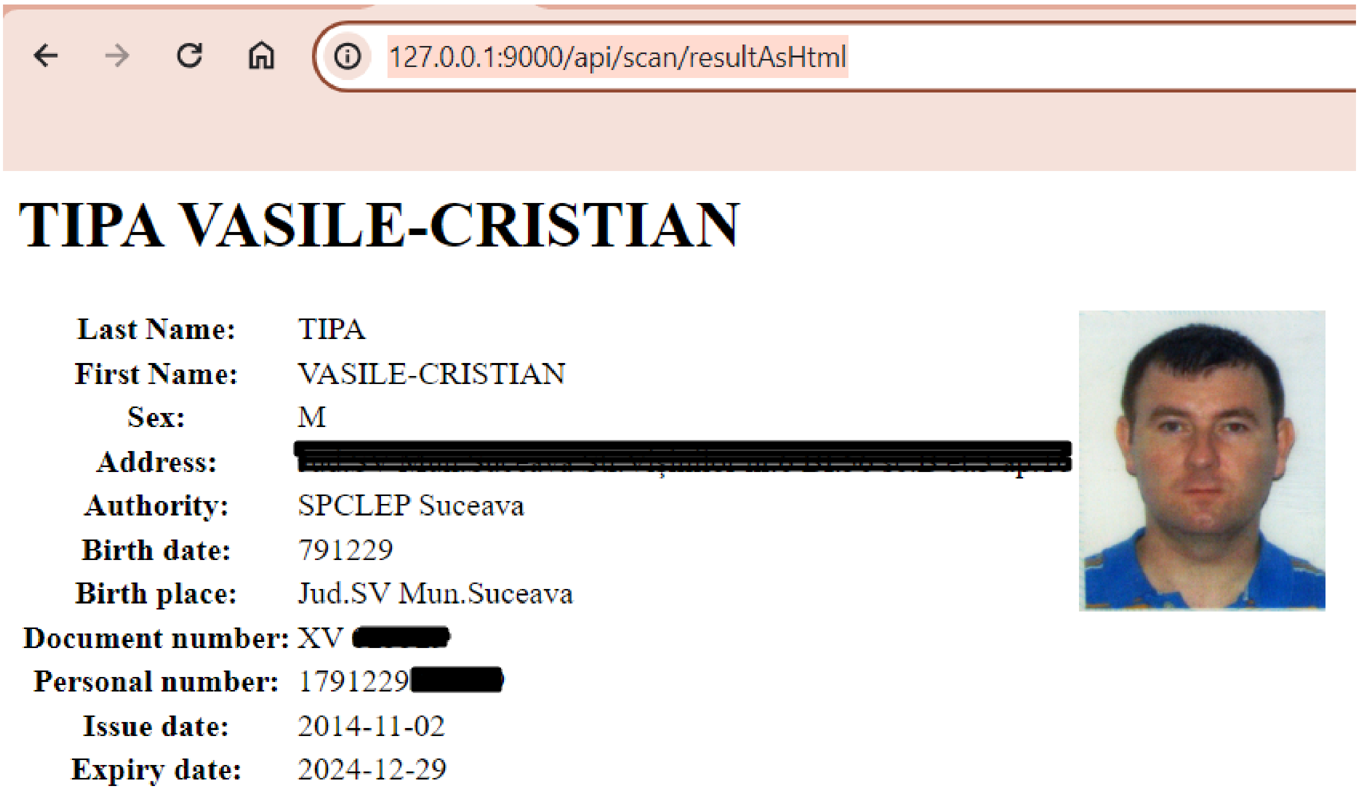
```
        None,
        Plugged,
```

scanState values are:

```
Idle

Scanning...

Classification failed! Unknown document

Scan completed

Gateway not licensed
```

http://127.0.0.1:9000/api/scan/resultAsHtml

This endpoint can be used after a scan to fetch the result in a user readable format. Does not contain all fields, but just some of them.



127.0.0.1:9000/api/scan/resultAsHtml

TIPA VASILE-CRISTIAN

Last Name:	TIPA
First Name:	VASILE-CRISTIAN
Sex:	M
Address:	[redacted]
Authority:	SPCLEP Suceava
Birth date:	791229
Birth place:	Jud.SV Mun.Suceava
Document number:	XV [redacted]
Personal number:	1791229[redacted]
Issue date:	2014-11-02
Expiry date:	2024-12-29

2. Signotec

http://localhost:9000/api/sign/status

This endpoint will provide information about the status of the signotec device and scan state.

The response is the `ScannerStatus` json.

http://localhost:9000/api/sign/start

This enpoint will start a signature detection session. Call this before client needs to start signing. The led on the device should turn green after this endpoint is called

The response is the `ScannerStatus` json.

http://localhost:9000/api/sign/restart

Calling this will issue a restart of the signature scan.

The response is the `ScannerStatus` json.

http://localhost:9000/api/sign/cancel

Calling this will cancel the sign scan process. Led on the device should turn orange.

The response is the `ScannerStatus` json.

http://localhost:9000/api/sign/confirm

http://localhost:9000/api/sign/confirm?options=1

These endpoints are used to confirm a signature. Call this ONLY after used finished a good signature. The options can be a combination of flags Ored together in an integer value. For example, passing optins=1 will add a timestamp to the image signature. Passing 4097 to options (which is actually 0x01 | `0x1000`) will add timestamp and also make the background transparent.

By default, if options is not passed at all, the signature image will be passed with transparent background.

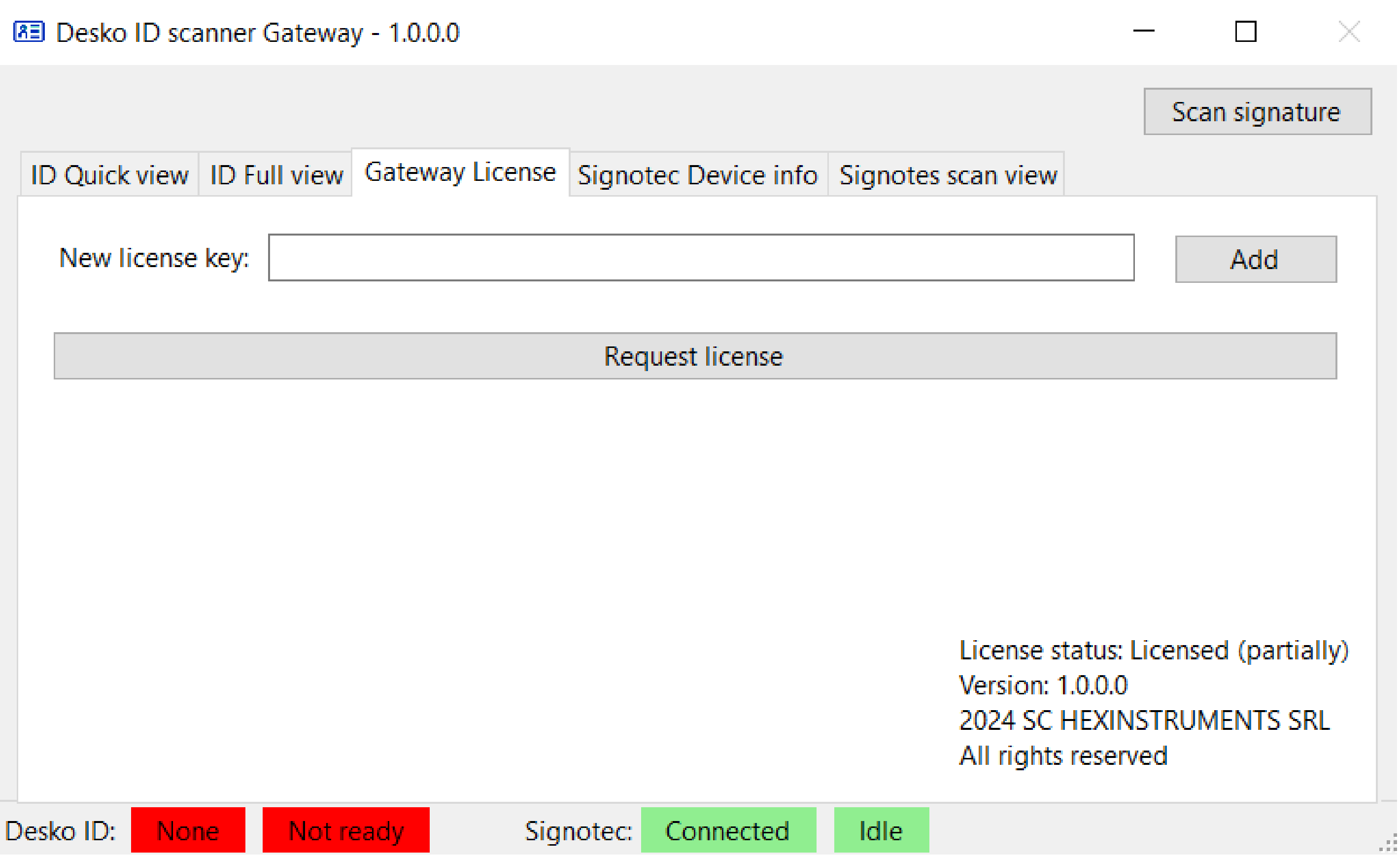
```
public enum SignatureImageFlag
    {
        None = 0x0,
        //
        // Summary:
        //     Include timestamp.
        Timestamp = 0x1,
        //
        // Summary:
        //     Include background image.
        BackImage = 0x2,
        //
        // Summary:
        //     Exlude the hot spot areas (fill with white color).
        ExcludeHotSpots = 0x4,
        //
        // Summary:
        //     Don't crop image to signature size.
        DontCrop = 0x8,
        //
        // Summary:
        //     Align signature on the left.
        AlignLeft = 0x10,
        //
        // Summary:
        //     Align signature on the right.
        AlignRight = 0x20,
        //
        // Summary:
        //     Align signature on the top.
        AlignTop = 0x40,
        //
        // Summary:
        //     Align signature on the bottom.
        AlignBottom = 0x80,
        //
        // Summary:
        //     Calculate timestamp size relative to the image size.
        TimestampRelToImage = 0x100,
        //
        // Summary:
        //     Never smooth signature.
        DontSmooth = 0x200,
        //
        // Summary:
        //     Always smooth signature.
        Smooth = 0x400,
        //
        // Summary:
        //     Include overlayed image.
        OverlayImage = 0x800,
        //
        // Summary:
        //     Draw the signatue on a transparent background.
        TransparentBack = 0x1000,
        //
        // Summary:
        //     Use current display content as background image.
        CurrentImages = 0x2000,
        //
        // Summary:
        //     The specified width is varied depending on the pressure values.
        VariablePenWidth = 0x4000
    }
```

The response of this endpoint is a FileContent with the image

Licensing Desko ID scanner Gateway

Desko ID scanner is not free and is allowed to be used only if licensed. The issued license for the gateway is per actual scanner device. Same instance of the Gateway can be licensed for multiple scanners but at a time only one of them will be usable from within the Gateway.

To get a license, go in the main window to the Gateway License tab and press the **Request license** button.



License status: Licensed (partially)
Version: 1.0.0.0
2024 SC HEXINSTRUMENTS SRL
All rights reserved

Desko ID: None Not ready Signotec: Connected Idle

After pressing this button, a file will be generated on disk in the application folder "licenseRequest.dat". Send this file to the license issuer and they will reply back with a license code that you will have to add in the **New license key** field and then press **Add** button.

By default, the application is licensed for the demo device and no license request is needed while working with this device.