Einführung

Diese API ist in der Lage, basierend auf einem von einem Fingerabdruck-Scanner generierten Hash Buchungen in TimeCard10 zu erstellen.

Voraussetzungen

Wichtig: Diese Pakete müssen vor der Ausführung dieser API installiert sein:

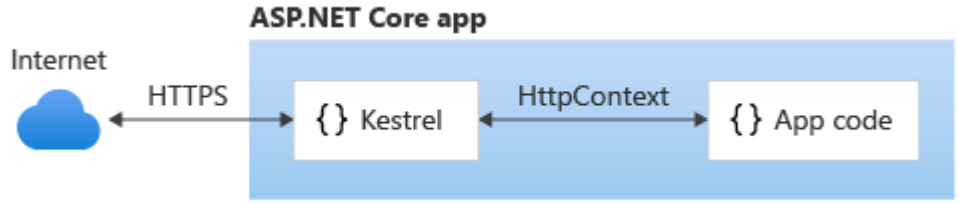
- .NET Core 6.0-Hosting-Bundle: https://dotnet.microsoft.com/en-us/download/dotnet/thank-you/runtime-aspnetcore-6.0.27-windows-hosting-bundle-installer

Hosting

Die API kann entweder direkt als „.exe“ oder über einen Reverse-Proxy gestartet werden. Bitte beachten Sie, dass die API standardmäßig nicht für die Verwendung eines SSL-Zertifikats konfiguriert ist. Daher sollte es nur von einem lokalen Netzwerk aus zugänglich sein oder hinter einem SSL-gesicherten Reverse-Proxy ausgeführt werden.

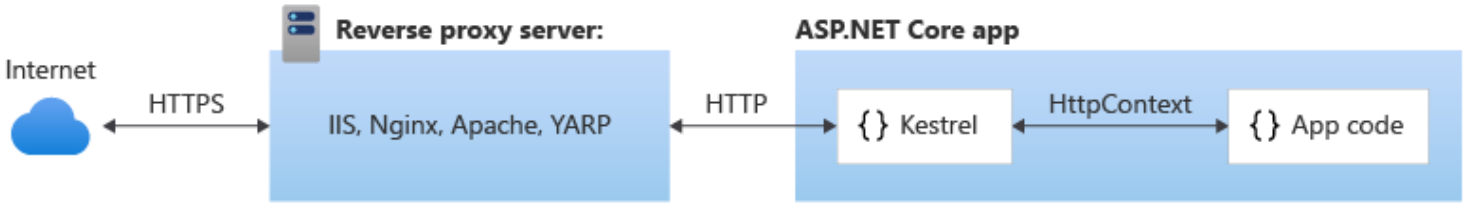
Ausführung als eigenständige `.exe`

Beim Öffnen der Anwendung `FingerprintAPI.exe` wird ein Kestrel-Webserver gestartet, der den in der Konfiguration definierten Port überwacht. Die API ist dann wie hier gezeigt zugänglich:



Ausführung hinter einem Reverse-Proxy

Das Ausführen einer API hinter einem Reverse-Proxy kann beim Lastausgleich, der Sicherheit und der Verwaltung der HTTPS-Einrichtung hilfreich sein.



Beispiel: Konfigurieren von Microsoft IIS als Reverse-Proxy

1. Installieren Sie das .NET Core Hosting Bundle (siehe Voraussetzungen).
2. Konfigurieren Sie den Anwendungspool
   - Öffnen Sie den IIS-Manager.
   - Klicken Sie im Bereich „Verbindungen“ mit der rechten Maustaste auf „Anwendungspools“ und wählen Sie „Anwendungspool hinzufügen“. – Geben Sie dem Anwendungspool einen Namen und legen Sie seine .NET CLR-Version auf „Kein verwalteter Code“ fest. .NET Core-Anwendungen werden in ihrem eigenen Prozess ausgeführt und sind nicht auf IIS angewiesen, um die .NET-Laufzeit zu verwalten.
   - Stellen Sie sicher, dass die Option „Anwendungspool sofort starten“ aktiviert ist.
3. Bereitstellung der API
   - Kopieren Sie die Anwendungsdateien in ein Verzeichnis auf dem Server. Es wird empfohlen, ein Verzeichnis zu verwenden, das im Ordner `C:\inetpub\` erstellt wurde, da die entsprechenden Berechtigungen in diesem Ordner von Microsoft bereits standardmäßig festgelegt sind. – Stellen Sie sicher, dass die Datei web.config im Stammverzeichnis der bereitgestellten Anwendung vorhanden ist. Diese Datei ist für das ASP.NET Core-Modul erforderlich, um Anforderungen zu konfigurieren und zu verarbeiten. Ein Beispiel ist unten dargestellt.
4. Erstellen Sie die Website
   - Klicken Sie im Bereich „Verbindungen“ mit der rechten Maustaste auf „Webseiten“ und wählen Sie „Webseite hinzufügen“.
   - Legen Sie einen aussagekräftigen Namen fest
   - Wählen Sie den zuvor erstellten Anwendungspool aus
   - Legen Sie den physischen Pfad fest, in dem Sie die API-Dateien bereitgestellt haben
   - Konfigurieren Sie die Bindung nach Bedarf (z. B. Portnummer, Hostname).

Konfigurieren Sie die Website so, dass sie als Reverse-Proxy für die API fungiert: Stellen Sie sicher, dass die Datei web.config im Stammverzeichnis der bereitgestellten Anwendung vorhanden ist. Diese Datei ist für das ASP.NET Core-Modul erforderlich, um Anforderungen zu konfigurieren und zu verarbeiten. Wenn die Datei fehlt, können Sie die unten gezeigte Konfiguration verwenden und sie in eine neue Datei mit dem Namen „web.config“ im Stammverzeichnis des Bereitstellungsordners einfügen.

Eine typische web.config würde so aussehen:

```
<configuration>
  <system.webServer>
    <handlers>
      <add name="aspNetCore" path="*" verb="*" modules="AspNetCoreModuleV2" resourceType="Unspeci
    </handlers>
    <aspNetCore processPath=".\FingerprintAPI.exe" stdoutLogEnabled="false" stdoutLogFile=".\logs
```

```
  </system.webServer>
</configuration>`
```

Konfiguration

Die Konfiguration wird im `JSON` -Format gespeichert. Die Parameter sind wie in der folgenden Beispielkonfiguration dargestellt strukturiert:

Beispielkonfiguration:

```
{
  "Logging": {
    "Level": "Information"
  },
  "Swagger": {
    "Path": ""
  },
  "WebApi": {
    "LicenseKey": "",
    "Port": 5011,
    "Security": {
      "Tokens": {
        "CreateBooking": "token123",
        "UpdatePersons": "token456",
        "ListDoors": "token789"
      }
    },
    "TimeCard10": {
      "BaseUrl": "https://timecard10.my-company.com",
      "Username": "",
      "Password": ""
    },
    "DoorOpener": {
      "BaseUrl": "https://timecard10.my-company.com",
      "Username": "",
      "Password": "",
      "PIN": ""
    },
    "FingerprintReaders": [
      {
        "Name": "Test 1",
        "IPAddresses": "0.0.0.1,127.0.0.1",
        "Location": {
          "Latitude": "49.2714379",
          "Longitude": "11.4656152"
        },
```

```
        "BookingComment": "Fingerprint Test 1",
        "Doors": [
          {
            "Id": 1,
            "Name": "Door 1",
            "OpenOnEntry": true,
            "OpenOnExit": false
          },
          {
            "Id": 2,
            "Name": "Door 2",
            "OpenOnEntry": false,
            "OpenOnExit": false
          }
        ]
      }
    ]
  }
}
```

Protokollierung

Parameter	Beschreibung	Standardwert
Level	Mindestprotokollierungsebene. Einer von „Verbose“, „Debug“, „Information“, „Warning“ und „Error“. Für Produktionsumgebungen empfehlen wir die Verwendung von „Information“	„Information“

Swagger

Parameter	Beschreibung	Standardwert
Path	Es ist möglich, den Pfad der Swagger-API-Dokumentation hier zu überschreiben	""

WebApi

Parameter	Beschreibung	Standardwert
LicenseKey	Ein Schlüssel (im UUID-Format) ist beim Schmer.IT-Vertriebsteam erhältlich	
Port	Ein numerischer Wert, bei dem sich der Listener registrieren soll	5011

WebApi:Sicherheit:Tokens

Parameter	Beschreibung
CreateBooking	Wenn auf eine nicht leere Zeichenfolge festgelegt, muss jede /booking/-Anfrage einen Token-Parameter mit demselben Wert in der Abfragezeichenfolge enthalten
UpdatePersons	Wenn auf eine nicht leere Zeichenfolge festgelegt, muss jede /personinfos/-Anfrage einen Token-Parameter mit demselben Wert in der Abfragezeichenfolge enthalten
ListDoors	Wenn auf eine nicht leere Zeichenfolge festgelegt, muss jede /doors/-Anfrage einen Token-Parameter mit demselben Wert in der Abfragezeichenfolge enthalten

WebApi:TimeCard10

Parameter	Beschreibung
BaseURL	TimeCard 10-Basis-URL (z. B. https://timecard10.my-company.com
Username	TimeCard 10-Benutzername für einen Benutzer mit erhöhten Rechten
Password	TimeCard 10 Passwort für den oben angegebenen Benutzernamen

WebApi:DoorOpener

Parameter	Description
BaseURL	TimeCard 10-Basis-URL (z. B. https://timecard10.my-company.com
Username	TimeCard 10-Benutzername für einen Benutzer mit der Berechtigung, Türen zu öffnen
Password	TimeCard 10 Passwort für den oben angegebenen Benutzer
PIN	Der Türöffne-PIN Code des Benutzers

WebApi:FingerprintReaders

Enthält ein Array [] von [Fingerprint Reader-Konfigurationen] (#webapi-fingerprintreader), wie unten gezeigt:

WebApi:FingerprintReader

Parameter	Beschreibung
Name	Name des Fingerabdrucklesers. Nur zu Informationszwecken, wird nicht verwendet
IPAddresses	Eine durch Kommas getrennte Liste von IPv4-IP-Adressen, die Zugriff auf diese API haben sollen. Wird die API nur im lokalen Netzwerk genutzt, muss zusätzlich die lokale IP-Adresse des Absenders eingetragen werden, andernfalls die Öffentliche. Sie können auch ein * einfügen, um alle Absender zuzulassen (nicht empfohlen)
Location	Standortobjekt wie unten gezeigt. Die hier definierten Koordinaten werden für alle von diesem Fingerabdruckleser gesendeten Buchungsanfragen verwendet
BookingComment	Dieser Buchungskommentar wird für alle Buchungsanfragen gespeichert, die von diesem Fingerabdruckleser gesendet werden
Doors	Array [] von Türkonfigurationen, wie unten gezeigt

WebApi:FingerprintReader:Location

Parameter	Beschreibung
Latitude	Standort Breitengrad als Dezimalzahl mit einem Punkt als Dezimaltrennzeichen (englisches Format)
Longitude	Standort Längengrad als Dezimalzahl mit einem Punkt als Dezimaltrennzeichen (englisches Format)

WebApi:FingerprintReader:Door

Parameter	Beschreibung	Beispiel
Name	Name der Tür. Nur zu Informationszwecken, wird nicht verwendet	"Tür 1"
Id	Die ID der zu öffnenden Tür (numerischer Wert) aus TimeCard 10	1
OpenOnEntry	Wenn auf true gesetzt, wird diese Tür geöffnet, wenn eine „Kommende“ Buchung erstellt wird.	true
OpenOnExit	Wenn auf true gesetzt, wird diese Tür geöffnet, wenn eine „Leaving“-Buchung erstellt wird.	false

Protokolle

Die Protokolle finden Sie im Unterordner „logs“ des Installationsverzeichnisses

Verwendung

Mitarbeiter aktualisieren

Senden Sie eine „POST“-Anfrage an http://ip:port/personinfos/?token=**token**

- **Token**: „UpdatePersons“-Token aus dem Sicherheitsabschnitt der appsettings.json

Der Request Body muss eine gültige `.csv` -Datei enthalten, die alle Personen und biometrischen Hashes enthält und vom Fingerabdruckleser exportiert wurde.

Beispiel `.csv` :

```
Sep=;
Name;Email;Phone Number 1;Phone Number 2;Phone Number 3;Virtual Number;Card 1;Card 2;PIN Code;Vir
"Test Person 1";"";;;;;;;;;"476beaaa-11a0-4c44-1b42-44cef887ec9c"
"Test Person 2";"";;;;;;;;;"536beaaa-14a0-1b44-5c22-6acef887ec3a"
```

Beispiel: http://127.0.0.1:5011/personinfos/?token=token456

Weitere Einzelheiten finden Sie unter http://127.0.0.1:5011/swagger/.

Neue Buchung erstellen

Senden Sie einen `POST` -Request an http://ip:port/booking/**bimetricHash**?token=**token**

- **biometrischerHash**: Derselbe Wert, der über den Request `/personinfos/` von oben importiert wurde
- **Token**: `CreateBooking` -Token aus dem Sicherheitsabschnitt der appsettings.json

Beispiel: http://127.0.0.1:5011/476beaaa-11a0-4c44-1b42-44cef887ec9c?token=token123

Weitere Einzelheiten finden Sie unter http://127.0.0.1:5011/swagger/.