



---

VoicePIN Fixed Password biometric  
voice verification system — user  
documentation

Title	VoicePIN Fixed Password biometric voice verification system — user documentation.
Version	0.3.3
Applies to program version	0.11.2

## AUTHOR

### VoicePIN.com Sp. z o.o.

Krakusa 11 St.

30-535 Krakow, Poland

Entered into the Register of Entrepreneurs of the National Court Register kept by the District Court of Kraków-Sródmieście, XI Commercial Department of the National Court Register under KRS no. 0000401198, share capital PLN 1,002,500.00, NIP (Tax ID) 676-244-96-70, REGON 122420779

Phone: +48 12 378-98-21

Fax: +48 12 398-42-93

E-mail: [info@voicepin.com](mailto:info@voicepin.com)

## DISCLAIMER

This document is constitutes a sole property of VoicePIN.com Sp. z o.o. This document can only be used by individuals authorised by VoicePIN.com and must not be disclosed to third parties. Any unauthorised person who obtained this document is obliged to immediately return it to VoicePIN.com Sp. z o.o at the address: Krakusa 11, 30-535 Kraków, Poland. Publication, copying, distribution, or other similar actions are legally prohibited under sanctions arising from respective legal regulations. This document and any accompanying annexes are to be treated as confidential.

# Table of contents

1.	VoicePIN – biometric voice verification system.....	4
1.1.	Application.....	4
1.2.	VoicePIN system features .....	5
1.3.	Fixed Password and Text Independent.....	6
1.4.	Glossary .....	7
2.	VoicePIN Fixed Password technical specification .....	8
2.1.	Components of the system .....	8
2.2.	VoicePIN Fixed Password architecture diagram .....	9
2.3.	Input files.....	10
3.	System installation .....	11
3.1.	System requirements.....	11
3.2.	Firewall configuration.....	12
3.3.	Linux Installation .....	13
3.4.	Windows Installation.....	15
3.5.	Database configuration .....	19
4.	Maintenance documentation .....	20
4.1.	Locating and launching the system components .....	20
4.2.	Changing server connection properties.....	21
4.3.	Archiving and updating the system .....	22
4.4.	Reporting errors.....	23
4.5.	API key configuration .....	24
5.	Description of API functionality of the VoicePIN Fixed Password system.....	25
5.1.	Voiceprint-related operations .....	25
5.2.	Password Group and UBM file management.....	31

# 1. VoicePIN – biometric voice verification system

## 1.1. Application

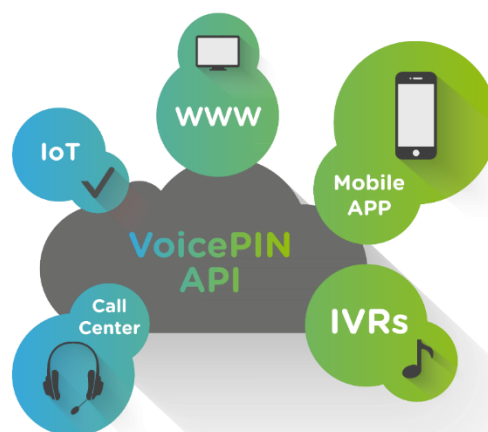
Conventional, widely used PIN or password based authorization method is not only uncomfortable for users, but also costly. Each connection with a hotline or application requires the user to recall or search for a password and enter it into the system.

Biometric voice verification not only allows to shorten the user authorization time by more than four times, but also increases the safety of this process. Contrary to the password, which can get into unauthorized hands, the voice is as unique as fingerprints or retina. This ensures that the clients' data are adequately protected and can be accessed in a quick and comfortable way.

Using the biometric "password" increases the company's image among customers as it contributes to the standard of customer service. State-of-the-art security systems and respecting the clients' time and comfort constitute an ideal marketing communication tool which distinguishes a company among others.

The VoicePIN system allows for a comfortable and reliable remote verification of clients via:

- phone (including GSM and VoIP),
- IVR system,
- web page,
- mobile application,
- interactive kiosk
- IoT devices.



### Note



The identity verification does not require remembering the password or PIN — the user's voice is the password.

## 1.2. VoicePIN system features

- verification of the correctness and integrity of test and training samples,
- checking the quality of test and training samples,
- smart adaptation of the system to new user voice samples,
- detecting unauthorized access attempts by sample analysis,
- high accuracy regardless of the language,
- support of several voice communication channels (GSM, PSTN, VoIP, Mobile, WEB),
- scalability according to the number of registered users and simultaneously used channels,
- integration with the existing computerized systems (hotlines, IVRs combined with ASR systems, conventional IVRs) via a dedicated API,
- application of valid application safety standards.

## 1.3. Fixed Password and Text Independent

The speaker biometric verification systems are basically divided into fixed password and text-independent systems, where:

- **text-independent systems** – [VoicePIN Text-Independent](#) – are easy to use during verification, but cannot ensure the required efficiency and safety in case of short phrases due to the voice changeability and the relatively long "training" sessions required. These systems are not well suited for maintenance-free commercial systems. They however work well during phone consultations by carrying out background verification, as they do not require a specific phrase to be pronounced.
- **fixed password systems** – [VoicePIN Fixed Password](#) – allow to significantly increase the efficiency and safety of biometric identity verification. Using fixed phrase during the "training" process allows to uniquely link the voice biometric model with phonemes of the password phrase – thus the system is sensitive to high level features such as intonation or accent. The enrollment, as well as verification, is much faster than in Text-Independent mode.

## 1.4. Glossary

- **UBM** – biometric universal background model for a given phrase; the system can determine which voice parameters in a given phrase are unique and which are common for a group of speakers,
- **Password Group** – groups the users using the same phrase within a given organization; an appropriate UBM file will be required to create a new Password Group,
- **Voiceprint** – a biometric statistic model of a user's voice represented by a matrix of parameters created on the basis of training recordings; a given Voiceprint can only be assigned to one Password Group only,
- **Voiceprint ID** – a unique Voiceprint ID number used during API method callouts which indicates the biometric model of a given user,
- **Playback Activity Detection (PAD)** – a system used to detect the re-use of a voice sample which has already been successfully used during a verification or an enrollment of the specified user
- **API key** – a unique programming interface access key which is generated during installation. Depending on the resources, Admin or Verifier type API key is required to call particular API methods.

## 2. VoicePIN Fixed Password technical specification

### 2.1. Components of the system

**VoicePIN Server** – an application which provides the biometric voice verification services by means of API using the REST Web Service and managing the system user models which are kept in a database. The programming interface is divided into:

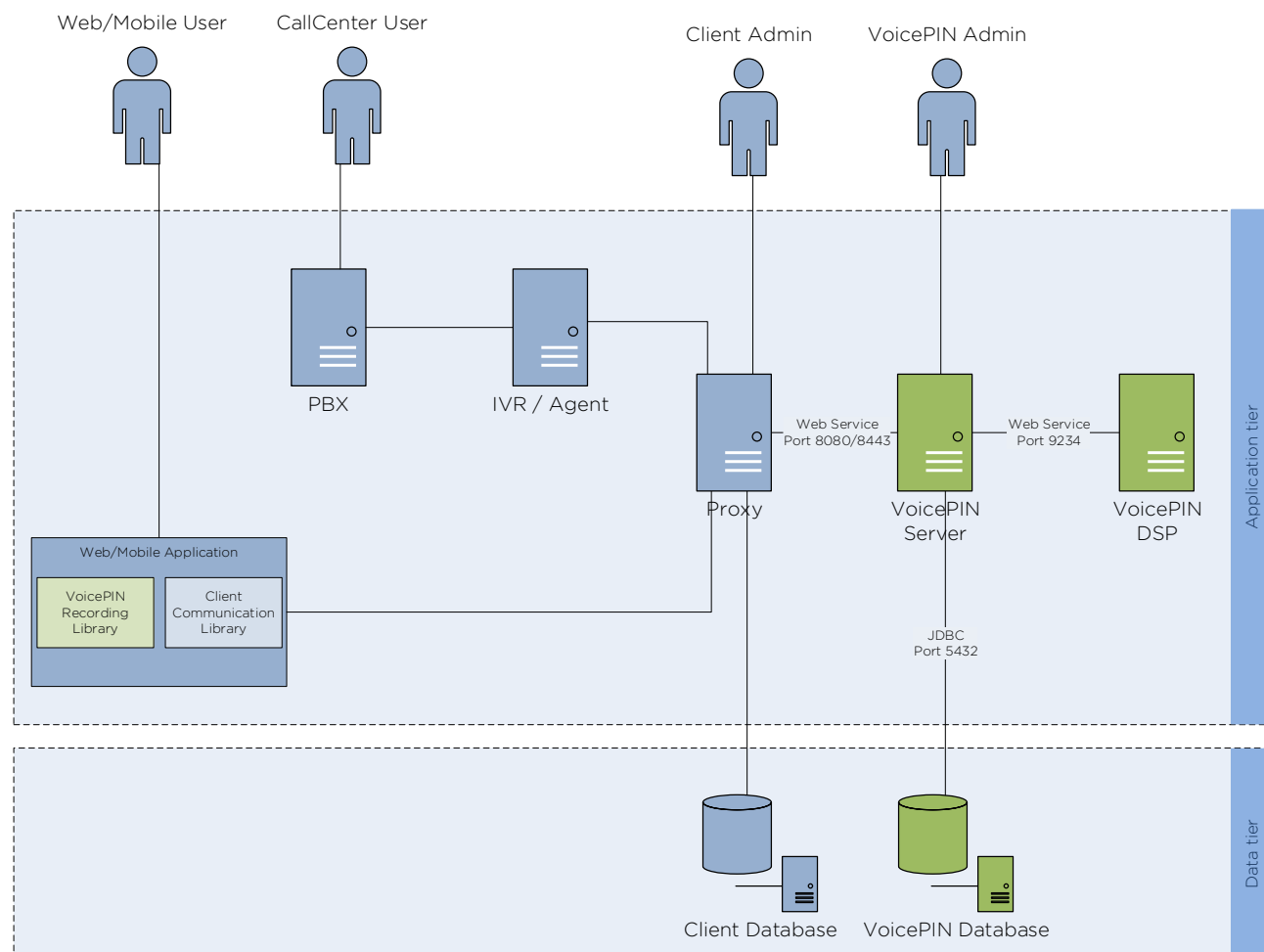
- API REST of biometric operations the access to which is authorized on the basis of **Verifier API key**,
- API REST for application management the access to which is authorized on the basis of **Admin API key**,




**VoicePIN DSP** – an engine used to process VoicePIN digital signals which communicates with VoicePIN Server directly via Web Service; it is inaccessible to external applications. It is a stateless application which allows for an easy scalability of the system by multiplication, depending on the number of users and verification frequency.

**VoicePIN Database** – a relational database storing user models, background statistic models for a given password, history of operations and system settings. PostgreSQL is the recommended database. Integration with other relational databases, such as MSSQL, is possible as well.



## 2.2.VoicePIN Fixed Password architecture diagram



Legend		
Logic architecture		
Symbol	Quantity	Description
	5	<b>Proxy</b> – Connection between VoicePIN verification system and clients systems like authorisation server, application backend and users' personal data <b>Web/Mobile Application</b> – Application (e.g. web or mobile) that collects users voice for authentication (may use VoicePIN libraries) and send it to the Proxy (connection as well as Proxy and application is not a part of VoicePIN system) <b>VoicePIN Server</b> – Application with programming interface (REST Web Services) that provides BVV service and manages user/background models <b>VoicePIN DSP</b> – Stateless digital signal processing VoicePIN engine that is a part of VoicePIN machine <b>PBX / IVR</b> – Call Center system elements that are not a part of VoicePIN system
	2	<b>VoicePIN Database</b> – Database of user models, background models and system configuration <b>Client Database</b> – Database of users' personal data
	4	<b>Mobile User</b> – End user of clients web/mobile application <b>CallCenter User</b> – End user that calls to the clients Call Center and is served by IVR or agent <b>VoicePIN Admin</b> – Administrator responsible for managing VoicePIN system <b>Client Admin</b> – Administrator responsible for managing personal data and security policy

## 2.3. Input files

The input file required for speaker enrollment or verification process is a

- WAVE Linear PCM recording.
  - Sampling frequency: 8000 Hz
  - Bit resolution: 16 bit
  - Bitrate: 128 kbps
  - No. of channels: 1 (mono)
  - Length: ~2 to 5 s
- WAVE U-Law (PCMU)
  - Sampling frequency: 8000 Hz
  - Bit resolution: 8 bit
  - Bitrate: 64 kbps
  - No. of channels: 1 (mono)
  - Length: ~2 to 5 s
- WAVE A-Law (PCMA)
  - Sampling frequency: 8000 Hz
  - Bit resolution: 8 bit
  - Bitrate: 64 kbps
  - No. of channels: 1 (mono)
  - Length: ~2 to 5 s

## 3. System installation

### 3.1. System requirements

#### Operating system

Centos 6.6/Oracle Enterprise Linux 6.6/RedHat 6/Windows Server 2012 Standard

#### Server hardware

**Assumption:** 100 simultaneous verifications within <3 s

No. of processors	2
Processor (no. of cores)	10
Processor (speed)	2.5 GHz
Processor (cache memory)	25 MB
RAM	32 GB DDR4
HDD	2 TB (2 x 1 TB RAID1)

## 3.2.Firewall configuration

### Note



Prior to the installation, check if the following ports are opened:

- 8080/TCP – in case of http protocol based connections,
- 8443/TCP – after configuring the connection by means of https protocol
- 5432/TCP – in case of remote database, on machine where database is installed

VoicePIN Server utilizes ports 9234/TCP for local communication with VoicePIN DSP and 5432/TCP for local communication with PostgreSQL database (if database is not remote).

### Checking for opened ports

Linux

```
$ iptables -L -n
```

Windows

```
telnet <ip_address> <port_number>
```

If “telnet” is not recognized as command, you have to go to Control Panel>Programs>Turn Windows features on or off. Then, check “Telnet Client” and save the changes.

If port is open, there will appear an empty window. If the port is closed, there will appear an error message.

### Opening ports

Linux

```
$ iptables -I INPUT -p tcp -m tcp --dport <port_number> -j ACCEPT  
$ service iptables save
```

Windows

```
netsh advfirewall firewall add rule name="<rule_name>" dir=in action=allow  
protocol=TCP localport=<port_number>
```

### 3.3.Linux Installation

#### Note



All commands listed below are to be carried out from the account with administrator access rights.

Apache Tomcat and PostgreSQL version will be set by the installation script. Database will be installed on the same machine with all other system components.

VoicePIN.tar.gz and install.sh files (available in the installation package) must be stored in the same location.

1. Grant rights to run **install.sh**:

```
$ sudo chmod +x install.sh
```

2. Launch the installation script:

```
$ ./install.sh
```

During the installation the user will be prompted to provide a password for voicepin database.

After the message "Installation has been done" appears, carry out the following:

3. Send **license.req** file from the `/usr/local/voicepin/` folder to [support@voicepin.com](mailto:support@voicepin.com) to enable a license to be generated.
4. After the VoicePIN Server license is provided, copy the license file to `/usr/local/voicepin/license/`. Make sure that the license file has read-only permission. If not, grant this permission:

```
$ chmod 644 license.crt
```

5. In the file `/etc/sudoers` comment the line:

```
Defaults requiretty
```

6. Make sure that the database is properly configured (see chapter 3.5) and restart the postgresql and tomcat services

```
$ service postgresql-9.3 restart
```

```
$ service tomcat start
```

7. In order to check whether the VoicePIN Server application was correctly started by Apache Tomcat, check the log file **voicepin.log** in the */usr/local/voicepin/logs/* folder or use App Manager application. At the end of the **voicepin.log** file the following line must be entered:

```
ContextLoader - Root WebApplicationContext: initialization completed in xxxx  
ms
```

## 3.4.Windows Installation

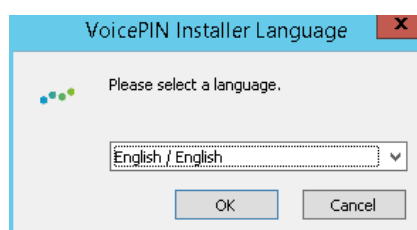
### Note



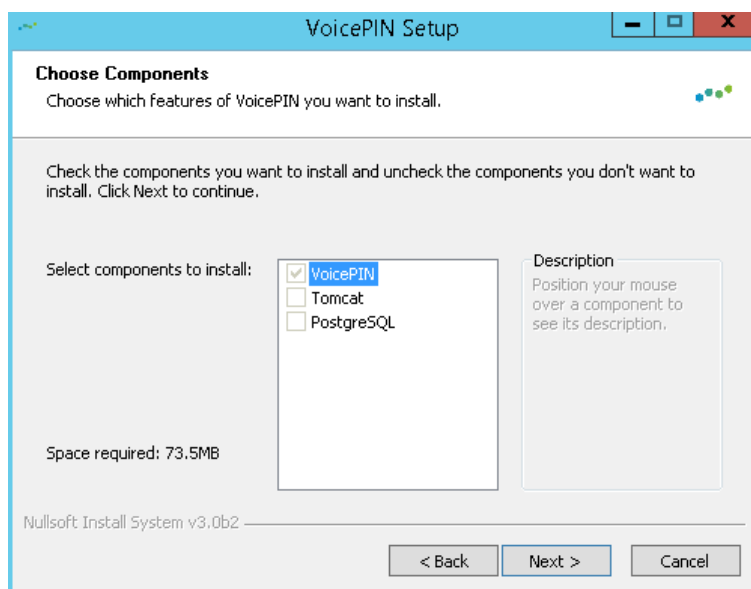
All commands listed below are to be carried out from the account with administrator access rights.

VoicePIN Fixed Password requires Java JRE version 1.7 or higher,

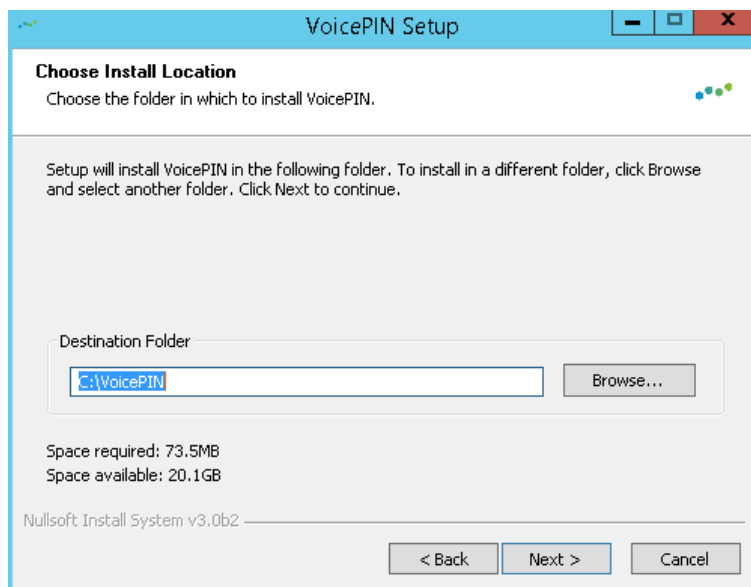
1. Make sure that Java JRE version 1.7 or higher is installed on the server.
2. Run *VoicePIN.exe* application with administrator access rights and select language of installation.



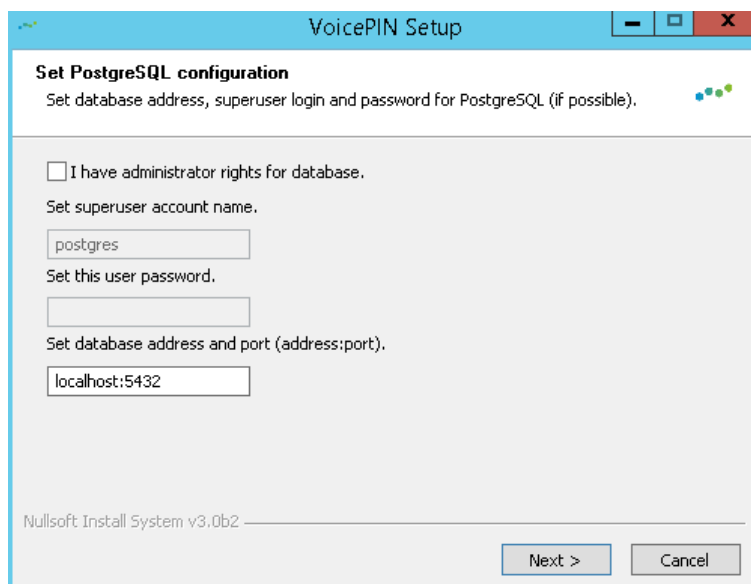
3. Select components to install. Note that VoicePIN Fixed Password requires Tomcat version 7 or higher and PostgreSQL version 9.2 or higher. Because system can communicate with remote database, installing PostgreSQL on a server is optional.



4. Choose where VoicePIN Fixed Password will be installed. Default destination folder is `C:\Program Files\VoicePIN`. If different location is selected, remember, that all files will not be stored in default location.



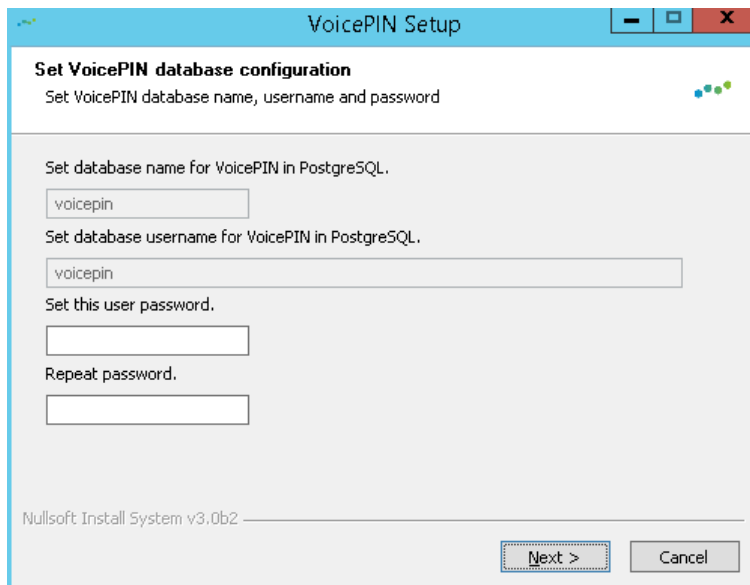
5. Select if you want to provide credentials for superuser in purpose of automatic creation of database and user dedicated to VoicePIN. If not, you will need to create VoicePIN database and user manually. If connecting to the remote database, remember to allow communication from the server ip address. Set the address and port to remote database.





- a. Superuser credentials were provided:

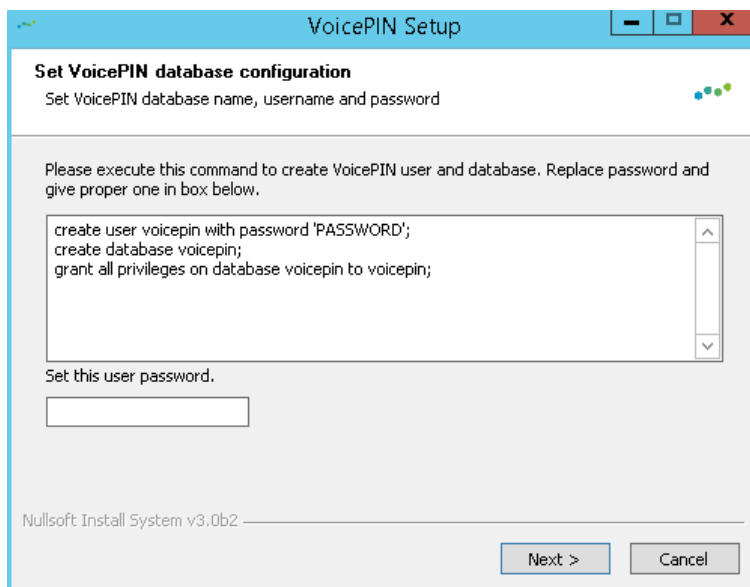
Set the password for VoicePIN database user.



The screenshot shows the 'VoicePIN Setup' window with the title 'Set VoicePIN database configuration'. Below the title is the instruction 'Set VoicePIN database name, username and password'. The window contains three input fields: 'Set database name for VoicePIN in PostgreSQL.' with 'voicepin' entered, 'Set database username for VoicePIN in PostgreSQL.' with 'voicepin' entered, and 'Set this user password.' with an empty field. Below the password field is a 'Repeat password.' field, also empty. At the bottom left, it says 'Nullsoft Install System v3.0b2'. At the bottom right, there are 'Next >' and 'Cancel' buttons.

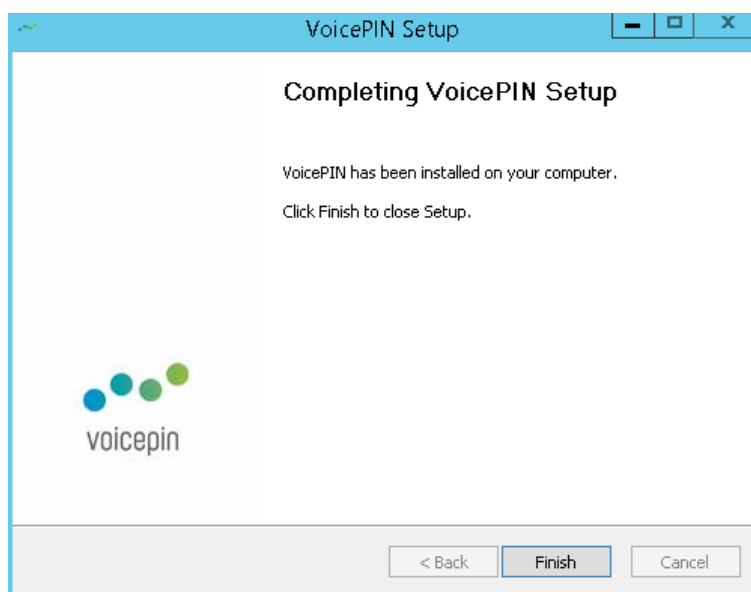
- b. Superuser credentials were not provided

Prepare database (only if superuser credentials were not provided) following the instructions from the installer window.



The screenshot shows the 'VoicePIN Setup' window with the title 'Set VoicePIN database configuration'. Below the title is the instruction 'Set VoicePIN database name, username and password'. The window contains a text area with the following SQL commands: 'create user voicepin with password 'PASSWORD';', 'create database voicepin;', and 'grant all privileges on database voicepin to voicepin;'. Below the text area is a 'Set this user password.' field, which is empty. At the bottom left, it says 'Nullsoft Install System v3.0b2'. At the bottom right, there are 'Next >' and 'Cancel' buttons.

6. Complete the installation process by clicking “Finish” button



7. In order to get valid license, send generated license request file (**license.req**) to [support@voicepin.com](mailto:support@voicepin.com). The default location of this file is *C:\Program Files\VoicePIN\VoicePIN-Server\*
8. After receiving a license, copy file **license.crt** to license folder (default location *C:\Program Files\VoicePIN\VoicePIN-Server\license\*)
9. Make sure that the database is properly configured (**see chapter 3.5**) and restart the postgresql and tomcat services

```
net stop postgresql-x64-<postgresql_version>
net start postgresql-x64-<postgresql_version>

net stop Tomcat7
net start Tomcat7
```

10. In order to check whether the VoicePIN Server application was correctly started by Apache Tomcat, check the log file **voicepin.log** in the *C:\Program Files\VoicePIN\VoicePIN-Server\logs\* folder or use App Manager application. At the end of the **voicepin.log** file the following line must be entered:

```
ContextLoader - Root WebApplicationContext: initialization completed in xxxx
ms
```

## 3.5.Database configuration

Files default location:

Linux:

**pg\_hba.conf** - `/var/lib/pgsql/<postgresql_version>/data/pg_hba.conf`

**postgresql.conf** - `/var/lib/pgsql/<postgresql_version>/data/postgresql.conf`

Windows:

**pg\_hba.conf** - `C:\Program Files\PostgreSQL\<postgresql_version>\data\pg_hba.conf`

**postgresql.conf** -

`C:\Program Files\PostgreSQL\<postgresql_version>\data\postgresql.conf`

### Local database

Open the file **pg\_hba.conf** and add line as below, as first entry in IPv4 local connections:

```
# IPv4 local connections:
host    <voicepin_db_user> <voicepin_database> 127.0.0.1/32    md5
host    all                all                127.0.0.1/32        ident
```

Open the file **postgresql.conf** and remove the comment (by deleting the first "#") from the line:

```
#port = 5432                                # (change requires restart)
```

### Remote database



Note

Changes are made to be made on remote machine, where database is installed.

Open the file **pg\_hba.conf** and add line:

```
host <voicepin_db_user> <voicepin_database> <voicepin_server_ip>/32    md5
```

Open the file **postgresql.conf** and remove the comment (by deleting the first "#") and from the line:

```
#port = 5432                                # (change requires restart)
```

In the same file uncomment line and add database server IP address after coma:

```
#listen_addresses = 'localhost,<database_server_IP>' # what IP address(es)
to listen on;
```

## 4. Maintenance documentation

### 4.1. Locating and launching the system components

#### VoicePIN Server

This application is launched together with the Tomcat application container.

Default Linux path:

*/usr/local/apache-tomcat-<tomcat\_version>/webapps/voicepin-server/*

Default Windows path:

*C:\Program Files\Apache Software Foundation\Tomcat  
<tomcat\_version>\webapps\voicepin-server\*

#### VoicePIN DSP

The DSP engine is launched as a service by the following command:

Linux:

```
$ service VoicePIN-DSP start
```

Default path:

*/usr/local/voicepin-dsp/*

Windows:

```
net start VoicePIN-DSP
```

Default path:

*C:\Program Files\VoicePIN\VoicePIN-DSP\*

## 4.2.Changing server connection properties

All system settings described in this chapter are provided in the **server.properties** configuration file.

Default location:

Linux: */usr/local/voicepin/conf/*

Windows: *C:\Program Files\VoicePIN\VoicePIN-Server\conf\*

### Database connection configuration

If the database settings are changed (e.g. password or username), the VoicePIN system settings must be changed accordingly. In this case, modify those properties:

```
persistence.dialect = SQLServer2008Dialect #SQL dialect
#supported SQL dialects: PostgreSQLDialect, SQLServer2008Dialect
persistence.host = <ip_address> #database server address
persistence.port = <port_number> #database server port
persistence.database = <database> #VoicePIN database name
persistence.user = <user> #username for VoicePIN database user
persistence.password = <password> #password for VoicePIN database user
```

### License path configuration

Default license file location may be changed using the following parameter

```
license.file = <license_path> #license file path
```

Remote locations are supported as long as they are available via file system (e.g. mapped network drive).

### VoicePIN DSP connection configuration

By default, VoicePIN-DSP is installed on the same machine as VoicePIN-Server and is communicating on local 9234 TCP port. However, there is an option to connect to remote DSP (e.g. for debugging purposes) – in this case, modify this line:

```
dsp.url = http://<ip_address>:<port_number> #voicepin-dsp connection url
```

## 4.3. Archiving and updating the system

### Updating the system

The system is updated by means of dedicated scripts provided by VoicePIN.com.

#### Note



Always adhere to instructions provided with each update.

### Archiving the system

In order to archive the system, export the copy of the existing database by means of a tool made available by the database provider.

## 4.4. Reporting errors

All system activities are saved in :

### VoicePIN-Server logs:

Recent log file: **voicepin.log**

Archive log files: **voicepin.log.<date>**

Default Linux location: */usr/local/voicepin/logs/*

Default Windows location: *C:\Program Files\VoicePIN\VoicePIN-Serv\logs\*

### VoicePIN-DSP logs:

Recent log file: **voicepin.log**

Archive log files: **voicepin.log.<date>**

Default Linux location: */usr/local/voicepin-dsp/logs/*

Default Windows location: *C:\VoicePIN\VoicePIN-DSP/logs/*

### Tomcat logs:

Recent log file: **catalina.out**

Archive log files: **catalina.out.<date> / catalina.<date>**

Default Linux location: */usr/local/apache-tomcat-<tomcat\_version>/log/*

Default Windows location: *C:\Tomcat<tomcat\_version>\logs\*

#### Note



If the system malfunction is detected, send the above mentioned files and the problem description to [support@voicepin.com](mailto:support@voicepin.com)

## 4.5.API key configuration

### Reading the API key

Both the Verifier API key and Admin API key values are required to communicate with the particular application resources and are generated during the application installation. To read the current API key value, run the proper script.

Linux:

To change user API key run **getApiKey.sh** script which default location is */usr/local/voicepin/conf/*

```
$ chmod +x getApiKey.sh
./getApiKey.sh
```

Windows:

Run **getApiKey.bat** script which default location is *C:\Program Files\VoicePIN\VoicePIN-Server\conf\*

### Configuring the API key

#### Note



The new API key must conform to the UUID (Universally Unique Identifier) requirements.

Admin API key must differ from the Verifier API key.

Changing the API key does not require the VoicePIN Server application to be restarted.

Linux:

To change user API key run **setUserApiKey.sh** script which default location is */usr/local/voicepin/conf/*

```
$ chmod +x setUserApiKey.sh
./setUserApiKey.sh
```

To change admin API key run **setAdminApiKey.sh** script which default location is */usr/local/voicepin/conf/*

```
$ chmod +x setAdminApiKey.sh
./setAdminApiKey.sh
```

Windows:

To change user API key run **setUserApiKey.bat** script which default location is *C:\Program Files\VoicePIN\VoicePIN-Server\conf\*

To change admin API key run **setAdminApiKey.bat** script which default location is *C:\Program Files\VoicePIN\VoicePIN-Server\conf\*



## 5. Description of API functionality of the VoicePIN Fixed Password system

### 5.1. Voiceprint-related operations

API enabling to carry out operations on Voiceprints is available at:

[http://<ip\\_address>:<tomcat\\_port>/voicepin-server/rest/v1/verifier/](http://<ip_address>:<tomcat_port>/voicepin-server/rest/v1/verifier/)

An interactive documentation of the described methods allowing to call them out from a web browser is available at:

[http://<ip\\_address>:<tomcat\\_port>/voicepin-server/](http://<ip_address>:<tomcat_port>/voicepin-server/)

#### Note



Verifier API key is required to call the methods related to Voiceprint management and use.

#### Note



All available API methods are described in **VoicePIN\_Verifier\_API\_V1.html** made available with the installation package.

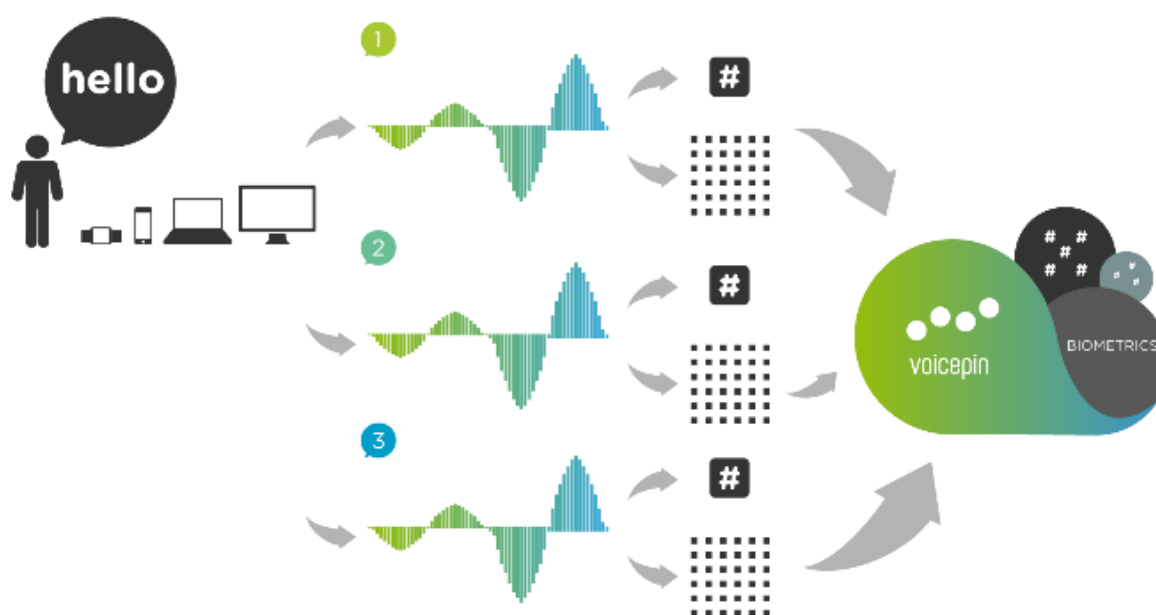
### Creating the Voiceprint

In order to carry out an operation with API VoicePIN for a given user, create his/her Voiceprint. To do so, call the POST method on the */passwordGroups/{passwordGroupName}* resource. From now on the returned Voiceprint ID will indicate the Voiceprint of a given user in the database. Until the registration, the Voiceprint cannot however be used for verification of a given speaker.

Voiceprint is assigned to a given Password Group (**passwordGroupName** parameter) which means that it is linked to the phrase which will be used during the verification.

## Registration (enrollment)

Before a user can be verified, the system must "know" his/her voice. To do so, the existing Voiceprint must be registered (so called "enrollment") on the basis of at least 3 voice samples which contain the phrase that will be used for verification. If a recording is unclear or has a noise in the background, the system may request another recording to be provided. During the registration a unique voice model is created which will be assigned to a given Voiceprint ID. In addition some characteristic features of the recording are registered which will be used by the PAD system.



*Fig. Voiceprint registration process*

To carry out the registration, call the POST method on the `/voiceprints/{voiceprintId}/enrollments` resource at least 3 times; thus the VoicePIN system will be provided with the phrase correctly pronounced by the same person. The person to be registered should speak naturally.

**enrollStatus** – system response that determines whether the recording was accepted or not; possible values:

- **SAMPLE\_ACCEPTED** – the recording is accepted,
- **SAMPLE\_REJECTED** – the recording contains an incorrect phrase or too much noise.

**trained** – information from the system returned after a sample was sent; determines whether a given Voiceprint is "trained" by appropriate number of samples.

- **false** – more samples are required,
- **true** – registration is successful.

The figure below shows a simplified diagram of system component integration during registration:

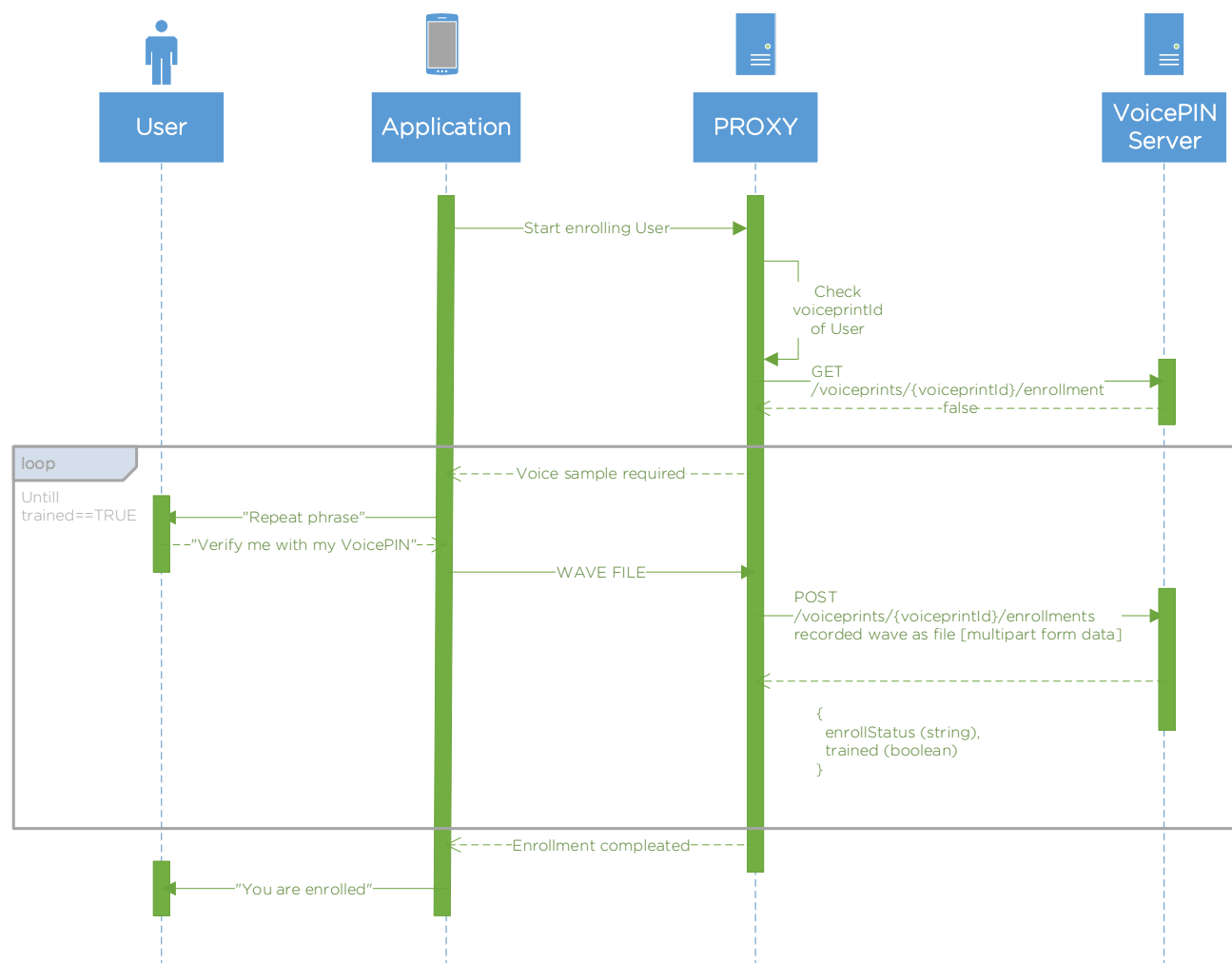


Fig. Integration of system components during registration

## Verification

If a Voiceprint is registered ("trained"), a biometric verification can be carried out; to do so, call the POST method on the `/voiceprint/{voiceprintId}/verifications` resource. This operation produces two values: *score* and *decision* which are described later in this document. These parameters can help to make a decision whether a given user should be authenticated, blocked or whether the verification attempt should be ignored.

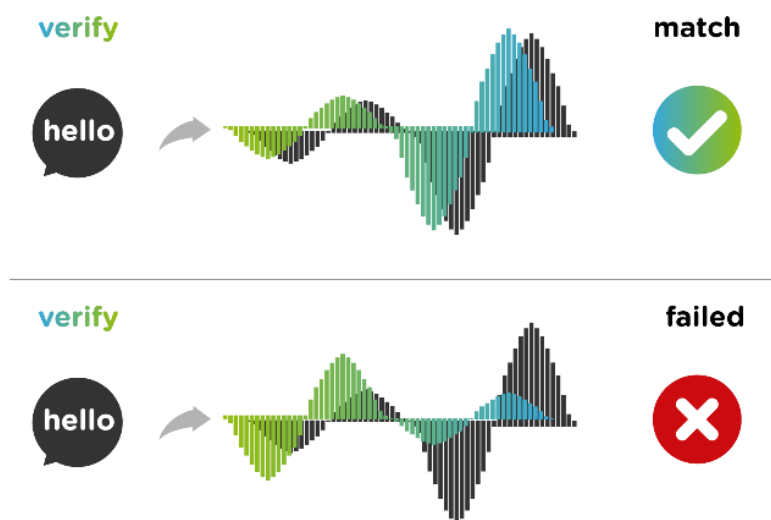


Fig.: Biometric verification process

**score** – the value returned by the system during the user verification. It represents the probability that the sample being verified and the statistical biometric model (Voiceprint) come from the same person. Higher values indicate higher reliability of the verification. The range of values depends on the phrase and the quality of system "training" – the better passphrase/training the higher scores will target users achieve. As standard, the *score* value for an attempt not being a Playback type attack ranges from  $\langle -50 \text{ to } 100 \rangle$ .

**decision** – the system's decision related to a given verification which is made on the basis of values obtained from the signal processing engine and threshold values for a given Password Group; possible values:

- MATCH – the identity is confirmed,
- MISMATCH – no biometric match,
- FRAUD – the re-use of a given sample was detected,
- WRONG\_PHRASE\_SPOKEN – the phrase is different from the one used during the registration.

The figure below shows a simplified diagram of system component integration during verification:

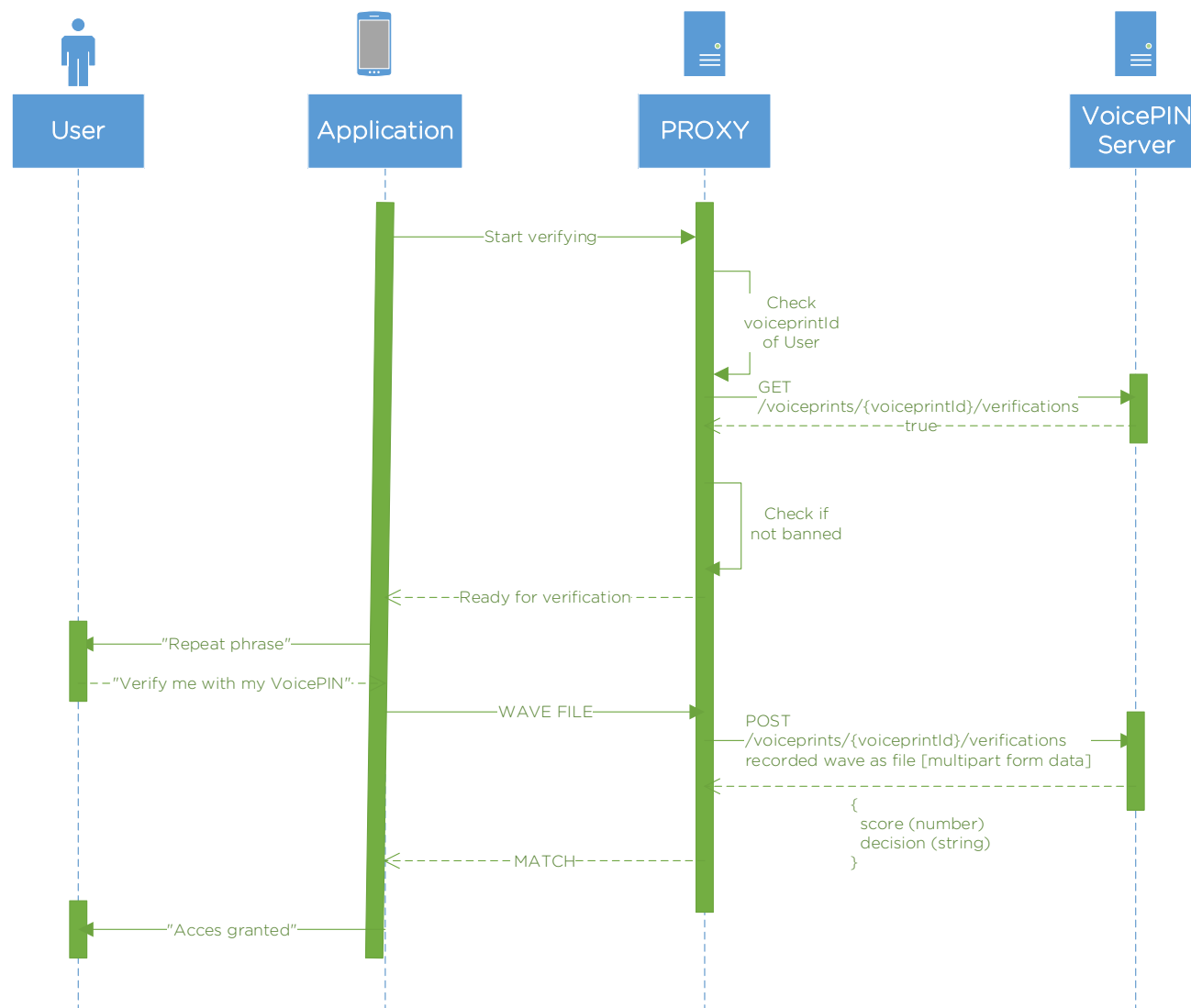


Fig.: Integration of system components during verification

## Resetting the Voiceprint

By calling out the DELETE method on the */voiceprints/{voiceprintId}/enrollments* resource, the system enables resetting a given Voiceprint. Resetting the Voiceprint can be required if a given user has problems with logging in due to noisy environment or a significant and permanent change of a transmission channel. After resetting, the Voiceprint must be registered again (enrollment).

### Note



Contrary to the Voiceprint removal, resetting does not cause the information required for PAD system operation to be removed from the database upon the re-registration.

## Voiceprint removal

In order to remove all information related to a given Voiceprint from the database, call the DELETE method on the */voiceprints/{voiceprintId}* resource.

### Note



Contrary to resetting the Voiceprint, the method described here removes the information required for correct operation of PAD system in case of re-registration.

## 5.2.Password Group and UBM file management

API allowing the Password Group and UBM file management is available at:

[http://<ip\\_address>:<tomcat\\_port>/voicepin-server/rest/v1/admin/](http://<ip_address>:<tomcat_port>/voicepin-server/rest/v1/admin/)

An interactive documentation of the described methods allowing to call them out from a web browser is available at:

[http://<ip\\_address>:<tomcat\\_port>/voicepin-server/?admin/](http://<ip_address>:<tomcat_port>/voicepin-server/?admin/)

### Note



Admin API key is required to call the methods related to Password Group and UBM file management.

### Note



All available API methods are described in `VoicePIN_Admin_API_V1.html` made available with the installation package.

### Importing a UBM file

In order to begin working with the phrase which was not available in the system, a UBM file must be imported. In other case an updated UBM file can be sent by a VoicePIN.com support team.

The UBM file can be imported by calling out a PUT method on the `/ubms/{ubmName}/import` resource. Apart from providing the path to the imported file, it is required to provide *ubmName* – a name under which a given file will be available in the database.

### Exporting a UBM file

It may be necessary to send a currently used UBM file to VoicePIN.com for diagnostic purposes or prior to the system update. In such case call the GET method on the `/ubms/{ubmName}/export` resource.

### Removing a UBM file

The system allows to delete an unused UBM file by calling out the DELETE method on the `/ubms/{ubmName}` resource. Assigning the UBM file to any Password Group will prevent this file from being removed from the system database.

### Creating a new Password Group

A new Password Group needs to be created when a new UBM file is used in order to add a new verification passphrase. In such case every Password Group will use a different UBM file.

In order to create a new Password Group, call the POST method on the `/passwordGroups` resource. The parameters required for the Password Group creation are described below.

**passwordGroupName** – the Password Group name used for calling out API methods related to that Password Group.

**ubmName** – the name of the UBM file linked to the Password Group.

**verifyTreshold** – a Score threshold determining when the verification can be regarded as credible within a given Password Group. The values range from <-50 to 100>; the default value is 0.

**lexMatchTreshold** – determines the system sensitivity during the lexical correctness assessment. This value indicates the expected percentage of false rejections of correct recordings. For example: 1 means that about 1% of correct samples were incorrectly rejected due to the lexical correctness issues. Increasing this parameter will increase the restrictions on the lexical cohesion. The values range from <0 to 50> and the default value is 1. The value can be changed to the nearest per mile (0.01). Setting the value of 0 will disable checking for the lexical correctness.

### Modification of parameters of the existing Password Group

It may be necessary to change the parameters of a given Password Group in order to increase the security or system's usability (modification of *verifyTreshold*) levels, change the verification mechanism sensitivity (modification of *lexMatchTreshold*) or select another UBM file.

To do so, call the PUT method on the */passwordGroups/{passwordGroupName}* resource and provide only those parameters as arguments which have to be modified. Selecting a new UBM file will cause all Voiceprints to be updated to the new UBM file. Although this process runs in the background, some users will be assigned to the previous version of the UBM file until all Voiceprints are updated.

### Removing a Password Group

To remove a Password Group, call the DELETE method on the */passwordGroups/{passwordGroupName}* resource.

#### Note



Calling this method for a given Password Group will cause all its Voiceprints to be removed together with any information related to these Voiceprints. Using this method is not recommended.