



**Professional Services**

**Nuance Integration Web Service**

**Kaiser Permanente**

**16th November, 2018**

# Copyright Information

©2018. SecureAuth® is a registered trademark of SecureAuth Corporation. SecureAuth’s IdP software, appliances, and other products and solutions are copyrighted products of SecureAuth Corporation.

Version 1.0

October 2017

# Table of Contents

[Copyright Information 2](#_Toc529443881)

[Table of Contents 3](#_Toc529443882)

[1. Introduction 4](#_Toc529443883)

[2. Document Usage 4](#_Toc529443884)

[3. Classic Platform Integration Overview 4](#_Toc529443885)

[4. Provisioning Platform Integration Overview 4](#_Toc529443886)

[5. Nuance Caller Processing Sequence 5](#_Toc529443887)

[Registration 5](#_Toc529443888)

[Password Reset 6](#_Toc529443889)

[6. User Registration Status 6](#_Toc529443890)

[Is User Registered Request 6](#_Toc529443891)

[Is User Registered Response 7](#_Toc529443892)

[7. PIN Validation 7](#_Toc529443893)

[Validate PIN Request 7](#_Toc529443894)

[Validate PIN Response 8](#_Toc529443895)

[8. Set Secret Date 8](#_Toc529443896)

[Set Secret Date Request 8](#_Toc529443897)

[Set Secret Date Response 9](#_Toc529443898)

[9. Authenticate User 9](#_Toc529443899)

[Authenticate User Request 9](#_Toc529443900)

[Authenticate User Response 10](#_Toc529443901)

[10. Password Reset 10](#_Toc529443902)

[Reset Password Request 10](#_Toc529443903)

[Reset Password Response 11](#_Toc529443904)

[11. Current v7.x ASP Interfaces 11](#_Toc529443905)

[12. Sample v7.x Dialog Trace 13](#_Toc529443906)

## 

# Introduction

The QuickFactor SOAP web service interface allows external applications to submit requests to the Access Assurance Suite (AAS) for processing.

# Document Usage

This document is intended to be used by SecureAuth and Nuance software developers for interfacing to the AAS system

# Classic Platform Integration Overview

This provides an overview of the current production solution on AAS v7.x



# Provisioning Platform Integration Overview

This provides an overview of the planned production solution on AAS v9.1.2



# Nuance Caller Processing Sequence

## Registration

1. **Is User Registered**
   1. Caller calls designated number
   2. Caller enters NUID
   3. Nuance checks ServiceNow for Caller status
      1. Caller must be ‘active’ to proceed
   4. Nuance checks registration status with AAS
      1. Registered: PIN=’disabled’
      2. Not Registered: PIN≠’disabled’
   5. If not registered, Caller is to record their voice print
2. **Validate PIN**
   1. Caller enters PIN
   2. Nuance requests AAS to verify the PIN
3. **Set Secret Date**
   1. Caller enters Secret Date
   2. Nuance requests AAS to store Secret Date
      1. AAS stores Secret Date (encrypted)
      2. AAS sets PIN=’disabled’
   3. Nuance records a hint for the Secret Date
   4. Nuance stores the voice recording directly in the database table ‘IVR\_Answers’
4. **Registration is complete.**

## Password Reset

1. **Is User Registered**
   1. Caller calls designated number
   2. Caller enters NUID
   3. Nuance checks ServiceNow for Caller status
      1. Caller must be ‘active’ to proceed
   4. Nuance checks registration status with AAS
      1. Registered: PIN=’disabled’
      2. Not Registered: PIN≠’disabled’
   5. If registered, Caller can proceed to reset their password
2. **Authenticate User**
   1. Caller enters their Secret Date
   2. Nuance requests AAS to authenticate Caller by Secret Date
      1. AAS returns set of password reset groups
3. **Password Reset**
   1. Caller selects password group to reset
   2. Nuance generates a random password
   3. Nuance requests AAS to reset the target group with the random password
   4. AAS checks profile Status=‘active’
      1. Return failure if not active
4. **Password reset is complete.**

# User Registration Status

This process will:

1. Verify the user profile status is ‘Active’
2. Check the PIN status (encrypted)
   1. Registered: PIN=’disabled’
   2. Not Registered: PIN≠’disabled’

## Is User Registered Request

URL: <URL>/users/{USERE\_ID}/checkregistration

Content-Type: application/JSON

Method: POST

Request Payload: NONE

## Is User Registered Response

##### User Already Registered

HTTP Status Code : 200

**Response Header**

Content-Type : application/json; charset=utf-8

Content-Length : the length of response

**Response Body:**

{

     "result": 0

}

##### User Not Registered Yet

HTTP Status Code : 200

**Response Header**

Content-Type : application/json; charset=utf-8

Content-Length : the length of response

**Response Body:**

{

     "result": 1

}

##### User Does not Exist

HTTP Status Code : 404

**Response Header**

Content-Type : application/json; charset=utf-8

Content-Length : the length of response

**Response Body:**

{

    "error": {

        "name": "EntityNotFound",

        "message": "Could not find any entity of type \"Users\" matching: 11"

    }

}

**Errors**

HTTP Status Code : 500

**Response Header**

Content-Type : application/json; charset=utf-8

Content-Length : the length of response

**Response Body:**

{

    "error": {

        "name": "ApiError",

        "message": "<Actual Error Message>"

    }

}

# PIN Validation

This process will:

1. Authenticate the user using the PIN for initial Nuance registration
   1. The PIN is encrypted in the SecureAuth database

## Validate PIN Request

URL: <URL>/users/{USERE\_ID}/validatepin

Content-Type: application/JSON

Method: POST

Request Payload

{

"pin":"1977"

}

## Validate PIN Response

##### User Already Registered

HTTP Status Code : 200

**Response Header**

Content-Type : application/json; charset=utf-8

Content-Length : the length of response

**Response Body:**

{

     "result": 0

}

##### User Not Registered and PIN matches

HTTP Status Code : 200

**Response Header**

Content-Type : application/json; charset=utf-8

Content-Length : the length of response

**Response Body:**

{

     "result": 1

}

##### User Not Registered and PIN does not match

HTTP Status Code : 200

**Response Header**

Content-Type : application/json; charset=utf-8

Content-Length : the length of response

**Response Body:**

{

     "result": 0

}

##### User Does not Exist

HTTP Status Code : 404

**Response Header**

Content-Type : application/json; charset=utf-8

Content-Length : the length of response

**Response Body:**

{

    "error": {

        "name": "EntityNotFound",

        "message": "Could not find any entity of type \"Users\" matching: 11"

    }

}

**Errors**

HTTP Status Code : 500

**Response Header**

Content-Type : application/json; charset=utf-8

Content-Length : the length of response

**Response Body:**

{

    "error": {

        "name": "ApiError",

        "message": "<Actual Error Message>"

    }

}

# Set Secret Date

This process will:

1. Set the Secret Date that the user registered via Nuance
   1. The Secret Date will be stored encrypted in the SecureAuth database
2. Set the PIN=’disabled’ (encrypted)

## Set Secret Date Request

URL: <URL>/users/{USERE\_ID}/secrets

Content-Type: application/JSON

Method: POST

Request Payload

{

"secretDate":"2020-03-11"

}

## Set Secret Date Response

##### User Already registered and Secret already set

HTTP Status Code : 200

**Response Header**

Content-Type : application/json; charset=utf-8

Content-Length : the length of response

**Response Body:**

{

    “result”:0

}

**User Not Registered yet**

HTTP Status Code : 200

**Response Header**

Content-Type : application/json; charset=utf-8

Content-Length : the length of response

**Response Body:**

{

    “result”:1

}

##### User Does not Exist

HTTP Status Code : 404

**Response Header**

Content-Type : application/json; charset=utf-8

Content-Length : the length of response

**Response Body:**

{

    "error": {

        "name": "EntityNotFound",

        "message": "Could not find any entity of type \"Users\" matching: 11"

    }

}

**Errors**

HTTP Status Code : 500

**Response Header**

Content-Type : application/json; charset=utf-8

Content-Length : the length of response

**Response Body:**

{

    "error": {

        "name": "ApiError",

        "message": "<Actual Error Message>"

    }

}