**Medical Device Voice Authentication**

**Josh Corbin, Devin Kloos, Kevin Copper, Patrick Donavan, Donald Taylor**

**Team 7 (Voice)  
April 28, 2017**

**ABSTRACT**

This project involves taking the requirements of a client who needs voice authentication for their medical device so that an authenticated doctor can gain access to the device to retrieve medical information pertaining to their patients. The goal is to provide effective, accurate, and secure means of authentication for the doctor to ensure that only the authorized personnel can gain access to the confidential data. This has been accomplished by utilizing the VoiceIt libraries, combined with a C++ wrapper. Upon implementation of the VoiceIt libraries, C++ wrapper, and our other customizations, we have created an effective voice authentication program to fulfill the client’s needs. Using this voice authentication software, healthcare settings can now securely utilize this medical device and ensure that only authorized individuals will be able to gain access to the confidential patient data to ensure strict adherence to HIPPA.

**TABLE OF CONTENTS**

[**INTRODUCTION**](#Introduction) **4**

[**REQUIREMENTS**](#Requirements) **5**

[**ANALYSIS**](#Analysis) **9**

[**DESIGN**](#Design) **11**

[**IMPLEMENTATION**](#Implementation) **19**

[**TESTING**](#Testing) **34**

[**CONCLUSION**](#Conclusion) **37**

[**REFERENCES**](#References) **38**

**INTRODUCTION**

A medical device which is constantly reading and storing confidential data from a patient must only be accessible by an authorized user – that patient’s doctor. The healthcare industry must adhere to strict policies and regulations such as HIPPA, and Team 7’s Voice Authentication software will do just that. The software was built utilizing the 5-stage design method: Requirements, Analysis, Design, Implementation, Testing, and we will cover all of these design workflows throughout this paper. By conducting interviews with our client, analyzing all of his needs, and then implementing the correct solution, you will see that Team 7’s Voice Authentication software for this medical device will cover all areas of effectiveness, usability, and security.

**REQUIREMENTS**

**Application Domain:** Health care, doctor’s office, hospital

**Need:** Voice detection is used to authenticate a user (doctor) so that he/she can read out the vitals, data, and information for that patient (from their medical device).

**Perform:** Doctor inputs voice into device for authentication. If the voice patterns match the user who has access privileges to that medical device, then authenticate that user and begin moving on to the next module (i.e. reading off vitals, data, etc).

**Users:** People using this software will be the doctor of that patient (HIPPA / privacy).

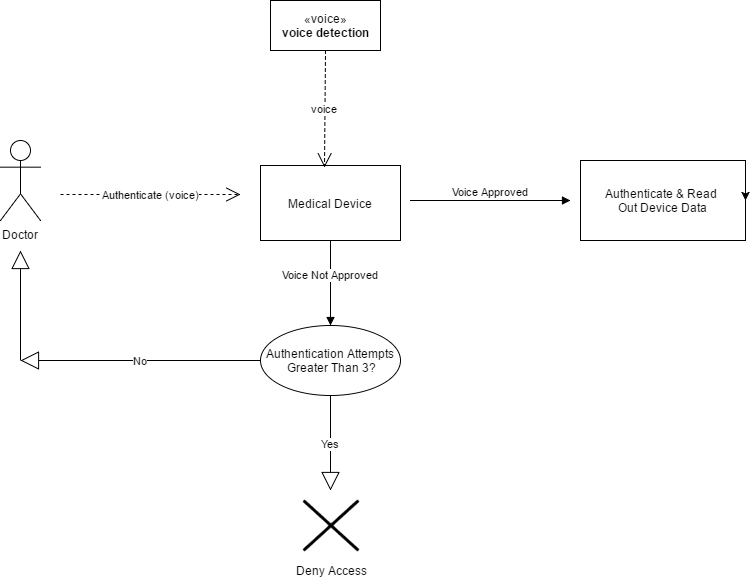
**Deadline:** April 27th, 2017

**Reliability:** Product must be operational 99% of the time or the mean time between failures must be at least 4 months.

**Is the software affordable / profitable:** Yes. Needed in health care industry.

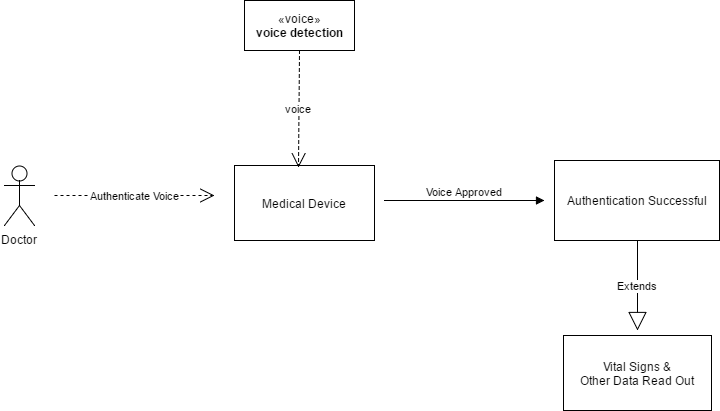
**Glossary:** "Confidential Information" means all business-related information, written or oral, that the Customer discloses or makes available to the Developer, directly or indirectly, through any means of communication or observation. "Software" means the programs created/used for the medical device voice detection system, and all associated documentation and other instructions.

**Business Model:** With the application domain defined, you can build a business model using UML diagrams to describe the client’s business processes. The business model is used to determine client’s initial requirements, then iteration.

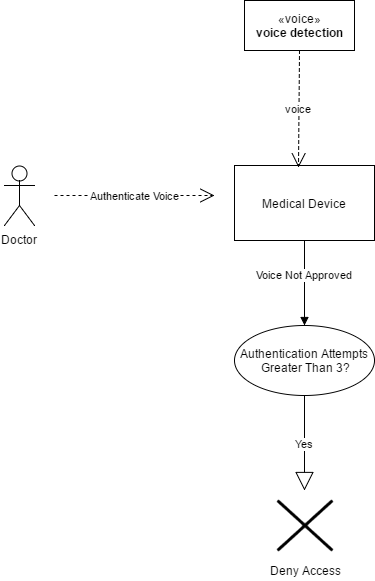
****

**Use Case models:** show interaction between software product and environment (i.e. Customer -> [banking software] -> Teller).

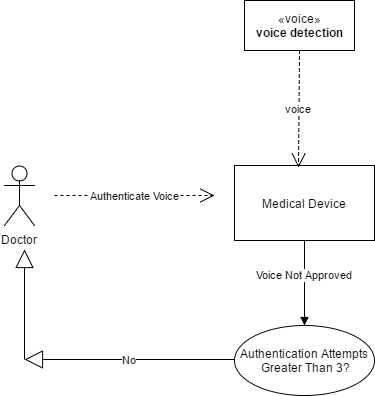
**Use Case 1 (Doctor Successfully Authenticated):**

****

**Use Case 2 (Authentication Failed 3+ Times(Access Denied)):**

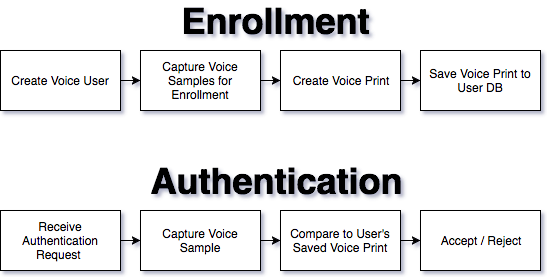
****

**Use Case 3 (Authentication Failed Less Than 3 Times (Reattempt Authentication)):**

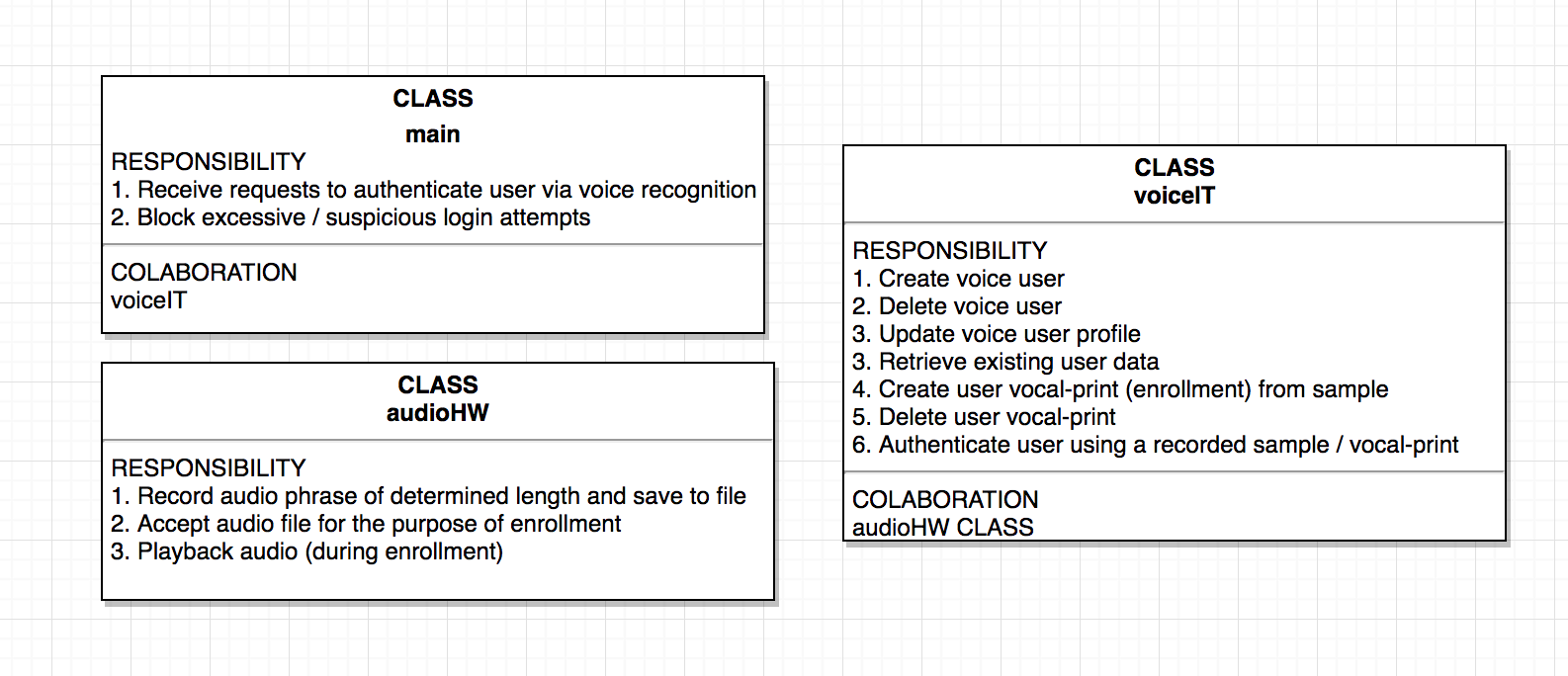
****

**ANALYSIS**

This is the basic process that must take place for voice authentication above.

****

From our layman term requirements I drew out this high level class requirements diagram.

****

This is the general data flow we can expect between the classes above.

****

**DESIGN**

**Class Diagram (UML)**

****

**Methods in Detail**

Class Name VoiceIt

Method name WriteCallback

Return type size\_t

Input argument(s) char \*contents, size\_t size, size\_t nmemb, void \*userp

Output argument(s) None

Error messages None

Files accessed None

Files changed None

Methods invoked readBuffer

Narrative

Method name read\_callback

Return type size\_t

Input argument(s) void \*ptr, size\_t size, size\_t nmemb, void \*userp

Output argument(s) None

Error messages None

Files accessed None

Files changed None

Methods invoked WriteThis, writeBuffer

Narrative

Method name getUser

Return type string

Input argument(s) string email, string password

Output argument(s) None

Error messages fprintf(stderr, "curl\_easy\_perform() failed: %s\n", curl\_easy\_strerror(res));

Files accessed None

Files changed None

Methods invoked readBuffer, CURL, CURLcode, curl\_easy\_strerror, curl\_easy\_perform, curl\_slist\_free\_all, curl\_easy\_cleanup, curl\_easy\_init, clear

Narrative

This REST API call is used to retrieve an existing user profile within the Voiceprint Developer Portal (VPDP) service. It retrieves an existing user profile record from the VPDP service database and returns true or false.

Method name setUser

Return type string

Input argument(s) string email,string password,string firstName,string lastName

Output argument(s) None

Error messages fprintf(stderr, "curl\_easy\_perform() failed: %s\n", curl\_easy\_strerror(res));

Files accessed None

Files changed None

Methods invoked readBuffer, CURL, CURLcode, curl\_easy\_strerror, curl\_easy\_perform, curl\_slist\_free\_all, curl\_global\_init, curl\_easy\_init, curl\_easy\_setopt, curl\_easy\_cleanup

Narrative

This REST API call is used to update an existing user profile within the Voiceprint Developer Portal (VPDP) service. It updates an existing user profile record within the VPDP service database and returns true or false.

Method name createUser

Return type string

Input argument(s) string email,string password,string firstName,string lastName, string phone1, string phone2, string phone3

Output argument(s) None

Error messages fprintf(stderr, "curl\_easy\_perform() failed: %s\n", curl\_easy\_strerror(res));

Files accessed None

Files changed None

Methods invoked CURL, CURLcode, curl\_easy\_strerror, curl\_easy\_perform, curl\_slist\_free\_all, curl\_global\_init, curl\_easy\_init, curl\_easy\_setopt, curl\_easy\_cleanup, readBuffer

Narrative:

Registers a new user profile within the Voiceprint Developer Portal (VPDP) service. Creates a new user profile record in the VPDP service database and returns true or false. Newly registered user profiles are enabled by default. Your DeveloperID is in the Welcome email you received when you registered.

Method name createUser

Return type string

Input argument(s) string email,string password,string firstName,string lastName

Output argument(s) email, password, firstName, lastName, "","",""

Error messages None

Files accessed None

Files changed None

Methods invoked createUser

Narrative

Registers a new user profile within the Voiceprint Developer Portal (VPDP) service. Creates a new user profile record in the VPDP service database and returns true or false. Newly registered user profiles are enabled by default. Your DeveloperID is in the Welcome email you received when you registered.

Method name deleteUser

Return type string

Input argument(s) string email,string password

Output argument(s) None

Error messages fprintf(stderr, "curl\_easy\_perform() failed: %s\n", curl\_easy\_strerror(res));

Files accessed None

Files changed None

Methods invoked curl\_easy\_strerror, curl\_easy\_perform, curl\_slist\_free\_all, curl\_global\_init, curl\_easy\_init, curl\_easy\_setopt, curl\_easy\_cleanup, readBuffer clear

Narrative

This REST API call is used to delete an existing user profile within the Voiceprint Developer Portal (VPDP) service. Deletes an existing user profile record from the VPDP service database and returns true or false.

Method name createEnrollment

Return type string

Input argument(s) string email,string passwrd, string pathToEnrollmentWav, string contentLanguage

Output argument(s) None

Error messages None

Files accessed fd

Files changed fd

Methods invoked clear, curl\_easy\_init, curl\_slist\_append, curl\_easy\_setopt, curl\_easy\_perform, curl\_slist\_free\_all, c\_str, curl\_easy\_cleanup, sha256, fopen, fstat, fileno

Narrative

This REST API call is used to create a new enrollment template for the specified user profile within the Voiceprint Developer Portal (VPDP) service. It creates a new enrollment template for the specified user profile in the VPDP service database and returns true or false. We recommend a minimum of three (3) enrollment templates per Voiceprint Phrase (VPP). Please Note: The Voiceprint Phrase's (VPP's) are Text-Dependent. The Minimum length of a VPP is 1.2 second. **Please note: You cannot use enrollment sound file for authentication. This is because of our anti- spoofing technology.** To manage the VPPs associated with your DeveloperID, please login to the developer portaland navigate to Voiceprint Phrases section. Please note: You can use DetectedVoiceprintText and DetectedTextConfidence to help decide which enrollmentID to keep or throw out and have the user record again based on text detected and its confidence.

Method name createEnrollment

Return type string

Input argument(s) string email,string passwrd, string pathToEnrollmentWav

Output argument(s) email,passwrd, pathToEnrollmentWav, ""

Error messages None

Files accessed fd

Files changed fd

Methods invoked createEnrollment

Narrative

This REST API call is used to create a new enrollment template for the specified user profile within the Voiceprint Developer Portal (VPDP) service. It creates a new enrollment template for the specified user profile in the VPDP service database and returns true or false. We recommend a minimum of three (3) enrollment templates per Voiceprint Phrase (VPP). Please Note: The Voiceprint Phrase's (VPP's) are Text-Dependent. The Minimum length of a VPP is 1.2 second. **Please note: You cannot use enrollment sound file for authentication. This is because of our anti- spoofing technology.** To manage the VPPs associated with your DeveloperID, please login to the developer portaland navigate to Voiceprint Phrases section. Please note: You can use DetectedVoiceprintText and DetectedTextConfidence to help decide which enrollmentID to keep or throw out and have the user record again based on text detected and its confidence.

Method name createEnrollmentByWavURL

Return type string

Input argument(s) string email,string passwrd, string urlToEnrollmentWav, string contentLanguage

Output argument(s) None

Error messages None

Files accessed None

Files changed None

Methods invoked CURL, CURLcode, clear, curl\_easy\_init, curl\_slist\_append, curl\_easy\_setopt, curl\_easy\_perform, curl\_slist\_free\_all, curl\_easy\_cleanup

Narrative

This REST API call is used to create a new enrollment template for the specified user profile within the Voiceprint Developer Portal (VPDP) service via a pre-recorded sound file. It creates a new enrollment template for the specified user profile in the VPDP service database and returns true or false. We recommend a minimum of three (3) enrollment templates per Voiceprint Phrase (VPP). Please Note: The Voiceprint Phrase's (VPP's) are Text-Dependent. The Minimum length of a VPP is 1.2 second. **Please note: You cannot use enrollment sound file for authentication. This is because of our anti- spoofing technology.** To manage the VPPs associated with your DeveloperID, please login to the developer portal and navigate to Voiceprint Phrases section. Please note: You can use DetectedVoiceprintText and DetectedTextConfidence to help decide which enrollmentID to keep or throw out and have the user record again based on text detected and its confidence.

Method name createEnrollmentByWavURL

Return type string

Input argument(s) string email,string passwrd, string urlToEnrollmentWav

Output argument(s) email,passwrd, urlToEnrollmentWav, ""

Error messages None

Files accessed None

Files changed None

Methods invoked createEnrollmentByWavURL

Narrative

This REST API call is used to create a new enrollment template for the specified user profile within the Voiceprint Developer Portal (VPDP) service via a pre-recorded sound file. It creates a new enrollment template for the specified user profile in the VPDP service database and returns true or false. We recommend a minimum of three (3) enrollment templates per Voiceprint Phrase (VPP). Please Note: The Voiceprint Phrase's (VPP's) are Text-Dependent. The Minimum length of a VPP is 1.2 second. **Please note: You cannot use enrollment sound file for authentication. This is because of our anti- spoofing technology.** To manage the VPPs associated with your DeveloperID, please login to the developer portal and navigate to Voiceprint Phrases section. Please note: You can use DetectedVoiceprintText and DetectedTextConfidence to help decide which enrollmentID to keep or throw out and have the user record again based on text detected and its confidence.

Method name getEnrollments

Return type string

Input argument(s) string email,string password

Output argument(s) None

Error messages fprintf(stderr, "curl\_easy\_perform() failed: %s\n", curl\_easy\_strerror(res));

Files accessed None

Files changed None

Methods invoked clear, curl\_easy\_init, curl\_slist\_append, curl\_easy\_setopt, curl\_easy\_perform, curl\_slist\_free\_all, c\_str

Narrative

This REST API call is used to retrieve the existing enrollment template(s) for the specified user profile within the Voiceprint Developer Portal (VPDP) service. It retrieves the existing enrollment template(s) for the specified user profile from the VPDP service database and returns true or false.

Method name deleteEnrollment

Return type string

Input argument(s) string email,string password,string enrollmentId

Output argument(s) None

Error messages None

Files accessed None

Files changed None

Methods invoked clear, curl\_easy\_init, curl\_slist\_append, curl\_easy\_setopt, curl\_easy\_perform, curl\_slist\_free\_all, c\_str

Narrative

This REST API call is used to delete an existing enrollment template for the specified user profile within the Voiceprint Developer Portal (VPDP) service. It deletes an existing enrollment template for the specified user profile from the VPDP service database and returns true or false.

Method name authentication

Return type string

Input argument(s) string email,string passwrd, string pathToAuthenticationWav, string confidence, string contentLanguage

Output argument(s) None

Error messages None

Files accessed None

Files changed None

Methods invoked clear, curl\_easy\_init, curl\_slist\_append, curl\_easy\_setopt, curl\_easy\_perform, curl\_slist\_free\_all, c\_str, curl\_easy\_cleanup, sha256, fopen, fstat, fileno

Narrative

This REST API call is used to authenticate the specified user profile within the Voiceprint Developer Portal (VPDP) service. It authenticates the specified user profile in the VPDP service database and returns success or failure. Please Note: The Voiceprint Phrase's (VPP's) are Text-Dependent. The Minimum length of a VPP is 1.5 second. **Please note: You cannot use enrollment sound file for authentication. This is because of our anti- spoofing technology.** To manage the VPPs associated with your DeveloperID, please login to the developer portaland navigate to Voiceprint Phrases section. We recommend starting with 85% for the confidence parameter during testing. After becoming familiar with the API, you can tweak the confidence parameter (with a maximum around 91) in order to decrease false positives. Please note: You can use DetectedVoiceprintText and DetectedTextConfidence to help decide which authentication to keep or throw out and have the user record again based on speech text detected and its confidence.

Method name authentication

Return type string

Input argument(s) string email,string passwrd, string pathToAuthenticationWav, string confidence

Output argument(s) email,passwrd,pathToAuthenticationWav,confidence,""

Error messages None

Files accessed None

Files changed None

Methods invoked authentication

Narrative

This REST API call is used to authenticate the specified user profile within the Voiceprint Developer Portal (VPDP) service. It authenticates the specified user profile in the VPDP service database and returns success or failure. Please Note: The Voiceprint Phrase's (VPP's) are Text-Dependent. The Minimum length of a VPP is 1.5 second. **Please note: You cannot use enrollment sound file for authentication. This is because of our anti- spoofing technology.** To manage the VPPs associated with your DeveloperID, please login to the [developer portal](https://siv.voiceprintportal.com/developerlogin.jsp)and navigate to Voiceprint Phrases section. We recommend starting with 85% for the confidence parameter during testing. After becoming familiar with the API, you can tweak the confidence parameter (with a maximum around 91) in order to decrease false positives. Please note: You can use DetectedVoiceprintText and DetectedTextConfidence to help decide which authentication to keep or throw out and have the user record again based on speech text detected and its confidence.

Method name authenticationByWavURL

Return type string

Input argument(s) string email,string passwrd, string urlToAuthenticationWav, string confidence, string contentLanguage

Output argument(s) None

Error messages None

Files accessed None

Files changed None

Methods invoked clear, curl\_easy\_init, curl\_slist\_append, curl\_easy\_setopt, curl\_easy\_perform, curl\_slist\_free\_all, c\_str, curl\_easy\_cleanup, sha256

Narrative

This REST API call is used to authenticate the specified user profile within the Voiceprint Developer Portal (VPDP) service via a pre-recorded sound file. It authenticates the specified user profile in the VPDP service database and returns success or failure. Please Note: The Voiceprint Phrase's (VPP's) are Text-Dependent. The Minimum length of a VPP is 1.2 second. **Please note: You cannot use enrollment sound file for authentication. This is because of our anti- spoofing technology.** To manage the VPPs associated with your DeveloperID, please login to the developer portal and navigate to Voiceprint Phrases section. We recommend starting with 85% for the confidence parameter during testing. After becoming familiar with the API, you can tweak the confidence parameter (with a maximum around 91) in order to decrease false positives. Please note: You can use DetectedVoiceprintText and DetectedTextConfidence to help decide which authentication to keep or throw out and have the user record again based on speech text detected and its confidence.

Method name authenticationByWavURL

Return type string

Input argument(s) string email,string passwrd, string urlToAuthenticationWav, string confidence

Output argument(s) email,passwrd,urlToAuthenticationWav,confidence,""

Error messages None

Files accessed None

Files changed None

Methods invoked authenticationByWavURL

Narrative

This REST API call is used to authenticate the specified user profile within the Voiceprint Developer Portal (VPDP) service via a pre-recorded sound file. It authenticates the specified user profile in the VPDP service database and returns success or failure. Please Note: The Voiceprint Phrase's (VPP's) are Text-Dependent. The Minimum length of a VPP is 1.2 second. **Please note: You cannot use enrollment sound file for authentication. This is because of our anti- spoofing technology.** To manage the VPPs associated with your DeveloperID, please login to the developer portaland navigate to Voiceprint Phrases section. We recommend starting with 85% for the confidence parameter during testing. After becoming familiar with the API, you can tweak the confidence parameter(with a maximum around 91) in order to decrease false positives. Please note: You can use DetectedVoiceprintText and DetectedTextConfidence to help decide which authentication to keep or throw out and have the user record again based on speech text detected and its confidence.

**IMPLEMENTATION**

Change 1: Implemented C++ Wrapper.

Change 2: Implemented Capture Modules.

**C++ Wrapper (voice.cpp)**

<https://github.com/kcopper/Team7_Voice/blob/master/voice.cpp>

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | #include <iostream> | |  | #include <string> | |  | #include "VoiceIt.h" | |  | using namespace std; | |  |  | |  | int main() { | |  | VoiceIt myVoiceIt("908e399927154baab125a5c2c9dc36af"); | |  | string createResponse = myVoiceIt.createUser("donavan.2@wright.edu","Pokemon1","Patrick","Donavan"); | |  | string enrollResponse = myVoiceIt.createEnrollment("donavan.2@wright.edu","Pokemon1", "rec\_4s.wav","en-US"); | |  | string authResponse = myVoiceIt.authentication("donavan.2@wright.edu","Pokemon1","rec\_4s.wav","90","en-US"); | |  |  | |  | return 0; | |  | } | |

**Entire Program (VoiceIt.h)**

<https://github.com/kcopper/Team7_Voice/blob/master/VoiceIt.h>

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | #ifndef \_VoiceIt\_\_ | |  | #define \_VoiceIt\_\_ | |  | #include <iostream> | |  | #include <fstream> | |  | #include <string> | |  | #include <cstring> | |  | #include <stdio.h> | |  | #include <vector> | |  | #include <curl/curl.h> | |  | #include <sys/stat.h> | |  | #include <fcntl.h> | |  | using namespace std; | |  |  | |  | class SHA256 | |  | { | |  | protected: | |  | typedef unsigned char uint8; | |  | typedef unsigned int uint32; | |  | typedef unsigned long long uint64; | |  |  | |  | const static uint32 sha256\_k[]; | |  | static const unsigned int SHA224\_256\_BLOCK\_SIZE = (512/8); | |  | public: | |  | void init(); | |  | void update(const unsigned char \*message, unsigned int len); | |  | void final(unsigned char \*digest); | |  | static const unsigned int DIGEST\_SIZE = ( 256 / 8); | |  |  | |  | protected: | |  | void transform(const unsigned char \*message, unsigned int block\_nb); | |  | unsigned int m\_tot\_len; | |  | unsigned int m\_len; | |  | unsigned char m\_block[2\*SHA224\_256\_BLOCK\_SIZE]; | |  | uint32 m\_h[8]; | |  | }; | |  |  | |  | std::string sha256(std::string input); | |  |  | |  | #define SHA2\_SHFR(x, n) (x >> n) | |  | #define SHA2\_ROTR(x, n) ((x >> n) | (x << ((sizeof(x) << 3) - n))) | |  | #define SHA2\_ROTL(x, n) ((x << n) | (x >> ((sizeof(x) << 3) - n))) | |  | #define SHA2\_CH(x, y, z) ((x & y) ^ (~x & z)) | |  | #define SHA2\_MAJ(x, y, z) ((x & y) ^ (x & z) ^ (y & z)) | |  | #define SHA256\_F1(x) (SHA2\_ROTR(x, 2) ^ SHA2\_ROTR(x, 13) ^ SHA2\_ROTR(x, 22)) | |  | #define SHA256\_F2(x) (SHA2\_ROTR(x, 6) ^ SHA2\_ROTR(x, 11) ^ SHA2\_ROTR(x, 25)) | |  | #define SHA256\_F3(x) (SHA2\_ROTR(x, 7) ^ SHA2\_ROTR(x, 18) ^ SHA2\_SHFR(x, 3)) | |  | #define SHA256\_F4(x) (SHA2\_ROTR(x, 17) ^ SHA2\_ROTR(x, 19) ^ SHA2\_SHFR(x, 10)) | |  | #define SHA2\_UNPACK32(x, str) \ | |  | { \ | |  | \*((str) + 3) = (uint8) ((x) ); \ | |  | \*((str) + 2) = (uint8) ((x) >> 8); \ | |  | \*((str) + 1) = (uint8) ((x) >> 16); \ | |  | \*((str) + 0) = (uint8) ((x) >> 24); \ | |  | } | |  | #define SHA2\_PACK32(str, x) \ | |  | { \ | |  | \*(x) = ((uint32) \*((str) + 3) ) \ | |  | | ((uint32) \*((str) + 2) << 8) \ | |  | | ((uint32) \*((str) + 1) << 16) \ | |  | | ((uint32) \*((str) + 0) << 24); \ | |  | } | |  |  | |  | const unsigned int SHA256::sha256\_k[64] = //UL = uint32 | |  | {0x428a2f98, 0x71374491, 0xb5c0fbcf, 0xe9b5dba5, | |  | 0x3956c25b, 0x59f111f1, 0x923f82a4, 0xab1c5ed5, | |  | 0xd807aa98, 0x12835b01, 0x243185be, 0x550c7dc3, | |  | 0x72be5d74, 0x80deb1fe, 0x9bdc06a7, 0xc19bf174, | |  | 0xe49b69c1, 0xefbe4786, 0x0fc19dc6, 0x240ca1cc, | |  | 0x2de92c6f, 0x4a7484aa, 0x5cb0a9dc, 0x76f988da, | |  | 0x983e5152, 0xa831c66d, 0xb00327c8, 0xbf597fc7, | |  | 0xc6e00bf3, 0xd5a79147, 0x06ca6351, 0x14292967, | |  | 0x27b70a85, 0x2e1b2138, 0x4d2c6dfc, 0x53380d13, | |  | 0x650a7354, 0x766a0abb, 0x81c2c92e, 0x92722c85, | |  | 0xa2bfe8a1, 0xa81a664b, 0xc24b8b70, 0xc76c51a3, | |  | 0xd192e819, 0xd6990624, 0xf40e3585, 0x106aa070, | |  | 0x19a4c116, 0x1e376c08, 0x2748774c, 0x34b0bcb5, | |  | 0x391c0cb3, 0x4ed8aa4a, 0x5b9cca4f, 0x682e6ff3, | |  | 0x748f82ee, 0x78a5636f, 0x84c87814, 0x8cc70208, | |  | 0x90befffa, 0xa4506ceb, 0xbef9a3f7, 0xc67178f2}; | |  |  | |  | void SHA256::transform(const unsigned char \*message, unsigned int block\_nb) | |  | { | |  | uint32 w[64]; | |  | uint32 wv[8]; | |  | uint32 t1, t2; | |  | const unsigned char \*sub\_block; | |  | int i; | |  | int j; | |  | for (i = 0; i < (int) block\_nb; i++) { | |  | sub\_block = message + (i << 6); | |  | for (j = 0; j < 16; j++) { | |  | SHA2\_PACK32(&sub\_block[j << 2], &w[j]); | |  | } | |  | for (j = 16; j < 64; j++) { | |  | w[j] = SHA256\_F4(w[j - 2]) + w[j - 7] + SHA256\_F3(w[j - 15]) + w[j - 16]; | |  | } | |  | for (j = 0; j < 8; j++) { | |  | wv[j] = m\_h[j]; | |  | } | |  | for (j = 0; j < 64; j++) { | |  | t1 = wv[7] + SHA256\_F2(wv[4]) + SHA2\_CH(wv[4], wv[5], wv[6]) | |  | + sha256\_k[j] + w[j]; | |  | t2 = SHA256\_F1(wv[0]) + SHA2\_MAJ(wv[0], wv[1], wv[2]); | |  | wv[7] = wv[6]; | |  | wv[6] = wv[5]; | |  | wv[5] = wv[4]; | |  | wv[4] = wv[3] + t1; | |  | wv[3] = wv[2]; | |  | wv[2] = wv[1]; | |  | wv[1] = wv[0]; | |  | wv[0] = t1 + t2; | |  | } | |  | for (j = 0; j < 8; j++) { | |  | m\_h[j] += wv[j]; | |  | } | |  | } | |  | } | |  |  | |  | void SHA256::init() | |  | { | |  | m\_h[0] = 0x6a09e667; | |  | m\_h[1] = 0xbb67ae85; | |  | m\_h[2] = 0x3c6ef372; | |  | m\_h[3] = 0xa54ff53a; | |  | m\_h[4] = 0x510e527f; | |  | m\_h[5] = 0x9b05688c; | |  | m\_h[6] = 0x1f83d9ab; | |  | m\_h[7] = 0x5be0cd19; | |  | m\_len = 0; | |  | m\_tot\_len = 0; | |  | } | |  |  | |  | void SHA256::update(const unsigned char \*message, unsigned int len) | |  | { | |  | unsigned int block\_nb; | |  | unsigned int new\_len, rem\_len, tmp\_len; | |  | const unsigned char \*shifted\_message; | |  | tmp\_len = SHA224\_256\_BLOCK\_SIZE - m\_len; | |  | rem\_len = len < tmp\_len ? len : tmp\_len; | |  | memcpy(&m\_block[m\_len], message, rem\_len); | |  | if (m\_len + len < SHA224\_256\_BLOCK\_SIZE) { | |  | m\_len += len; | |  | return; | |  | } | |  | new\_len = len - rem\_len; | |  | block\_nb = new\_len / SHA224\_256\_BLOCK\_SIZE; | |  | shifted\_message = message + rem\_len; | |  | transform(m\_block, 1); | |  | transform(shifted\_message, block\_nb); | |  | rem\_len = new\_len % SHA224\_256\_BLOCK\_SIZE; | |  | memcpy(m\_block, &shifted\_message[block\_nb << 6], rem\_len); | |  | m\_len = rem\_len; | |  | m\_tot\_len += (block\_nb + 1) << 6; | |  | } | |  |  | |  | void SHA256::final(unsigned char \*digest) | |  | { | |  | unsigned int block\_nb; | |  | unsigned int pm\_len; | |  | unsigned int len\_b; | |  | int i; | |  | block\_nb = (1 + ((SHA224\_256\_BLOCK\_SIZE - 9) | |  | < (m\_len % SHA224\_256\_BLOCK\_SIZE))); | |  | len\_b = (m\_tot\_len + m\_len) << 3; | |  | pm\_len = block\_nb << 6; | |  | memset(m\_block + m\_len, 0, pm\_len - m\_len); | |  | m\_block[m\_len] = 0x80; | |  | SHA2\_UNPACK32(len\_b, m\_block + pm\_len - 4); | |  | transform(m\_block, block\_nb); | |  | for (i = 0 ; i < 8; i++) { | |  | SHA2\_UNPACK32(m\_h[i], &digest[i << 2]); | |  | } | |  | } | |  |  | |  | std::string sha256(std::string input) | |  | { | |  | unsigned char digest[SHA256::DIGEST\_SIZE]; | |  | memset(digest,0,SHA256::DIGEST\_SIZE); | |  |  | |  | SHA256 ctx = SHA256(); | |  | ctx.init(); | |  | ctx.update( (unsigned char\*)input.c\_str(), input.length()); | |  | ctx.final(digest); | |  |  | |  | char buf[2\*SHA256::DIGEST\_SIZE+1]; | |  | buf[2\*SHA256::DIGEST\_SIZE] = 0; | |  | for (int i = 0; i < SHA256::DIGEST\_SIZE; i++) | |  | sprintf(buf+i\*2, "%02x", digest[i]); | |  | return std::string(buf); | |  | } | |  |  | |  | static string readBuffer; | |  |  | |  | class VoiceIt{ | |  | private: | |  | string developerId= ""; | |  | string urlUsers= "https://siv.voiceprintportal.com/sivservice/api/users"; | |  | string urlEnrollments = "https://siv.voiceprintportal.com/sivservice/api/enrollments"; | |  | string urlAuthentication = "https://siv.voiceprintportal.com/sivservice/api/authentications"; | |  |  | |  |  | |  |  | |  | public: | |  | VoiceIt(string devId){ | |  | cout<<"Constructor for VoiceIt Called"<<endl; | |  | developerId=devId; | |  | cout<<"Parameters Initialized"<<endl; | |  | } | |  |  | |  | static size\_t writeCallBack(char \*contents, size\_t size, size\_t nmemb, void \*userp) | |  | { | |  | size\_t realsize = size \* nmemb; | |  | readBuffer.append(contents, realsize); | |  | return realsize; | |  | } | |  |  | |  |  | |  |  | |  | struct WriteThis | |  | { | |  | const char \*readptr; | |  | long sizeleft; | |  | }; | |  |  | |  | static size\_t readCallBack(void \*ptr, size\_t size, size\_t nmemb, void \*userp) | |  | { | |  | struct WriteThis \*writeBuffer = (struct WriteThis \*)userp; | |  |  | |  | if(size\*nmemb < 1) | |  | return 0; | |  |  | |  | if(writeBuffer->sizeleft) { | |  | \*(char \*)ptr = writeBuffer->readptr[0]; /\* copy one single byte \*/ | |  | writeBuffer->readptr++; /\* advance pointer \*/ | |  | writeBuffer->sizeleft--; /\* less data left \*/ | |  | return 1; /\* we return 1 byte at a time! \*/ | |  | } | |  |  | |  | return 0; /\* no more data left to deliver \*/ | |  | } | |  |  | |  | string getUser(string email,string password) | |  | { | |  | CURL \*curl; | |  | CURLcode res; | |  | string emailParam = "VsitEmail: "+email; | |  | string passParam = "VsitPassword: "+sha256(password); | |  | string devIdParam = "VsitDeveloperId: "+developerId; | |  | string platformIdParam = "PlatformID: 11"; | |  | readBuffer.clear(); | |  | curl = curl\_easy\_init(); | |  | if(curl) { | |  | struct curl\_slist \*chunk = NULL; | |  | chunk = curl\_slist\_append(chunk, emailParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, passParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, devIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, platformIdParam.c\_str()); | |  | curl\_easy\_setopt(curl, CURLOPT\_URL, "https://siv.voiceprintportal.com/sivservice/api/users"); | |  | res = curl\_easy\_setopt(curl, CURLOPT\_HTTPHEADER, chunk); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEFUNCTION, writeCallBack); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEDATA, &readBuffer); | |  | /\* Perform the request, res will get the return code \*/ | |  | res = curl\_easy\_perform(curl); | |  | /\* Check for errors \*/ | |  | if(res != CURLE\_OK) | |  | fprintf(stderr, "curl\_easy\_perform() failed: %s\n", | |  | curl\_easy\_strerror(res)); | |  | /\* always cleanup \*/ | |  | curl\_easy\_cleanup(curl); | |  | //cout<<"Response is "<<readBuffer<<" For Sure"<<endl; | |  |  | |  | /\* free the custom headers \*/ | |  | curl\_slist\_free\_all(chunk); | |  | return readBuffer; | |  | } | |  | } | |  |  | |  | string setUser(string email,string password,string firstName,string lastName) | |  | { | |  | CURL \*curl; | |  | CURLcode res; | |  | string emailParam = "VsitEmail: "+email; | |  | string passParam = "VsitPassword: "+sha256(password); | |  | string devIdParam = "VsitDeveloperId: "+developerId; | |  | string fNameParam = "VsitFirstName: "+firstName; | |  | string lNameParam = "VsitLastName: "+lastName; | |  | string platformIdParam = "PlatformID: 11"; | |  | curl\_global\_init(CURL\_GLOBAL\_ALL); | |  | readBuffer.clear(); | |  | curl = curl\_easy\_init(); | |  | if(curl) { | |  | struct curl\_slist \*chunk = NULL; | |  | chunk = curl\_slist\_append(chunk, emailParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, passParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, devIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, fNameParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, lNameParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, platformIdParam.c\_str()); | |  | curl\_easy\_setopt(curl, CURLOPT\_CUSTOMREQUEST, "PUT"); | |  | curl\_easy\_setopt(curl, CURLOPT\_URL, "https://siv.voiceprintportal.com/sivservice/api/users"); | |  | res = curl\_easy\_setopt(curl, CURLOPT\_HTTPHEADER, chunk); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEFUNCTION, writeCallBack); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEDATA, &readBuffer); | |  | /\* Perform the request, res will get the return code \*/ | |  | res = curl\_easy\_perform(curl); | |  | /\* Check for errors \*/ | |  | if(res != CURLE\_OK) | |  | fprintf(stderr, "curl\_easy\_perform() failed: %s\n", | |  | curl\_easy\_strerror(res)); | |  | /\* always cleanup \*/ | |  | curl\_easy\_cleanup(curl); | |  | //cout<<"Response is "<<readBuffer<<" For Sure"<<endl; | |  | /\* free the custom headers \*/ | |  | curl\_slist\_free\_all(chunk); | |  | return readBuffer; | |  | } | |  | } | |  |  | |  | string createUser(string email,string password,string firstName,string lastName, string phone1, string phone2, string phone3) | |  | { | |  | CURL \*curl; | |  | CURLcode res; | |  | string emailParam = "VsitEmail: "+email; | |  | string passParam = "VsitPassword: "+sha256(password); | |  | string devIdParam = "VsitDeveloperId: "+developerId; | |  | string fNameParam = "VsitFirstName: "+firstName; | |  | string lNameParam = "VsitLastName: "+lastName; | |  | string phone1Param = "VsitPhone1: "+phone1; | |  | string phone2Param = "VsitPhone2: "+phone2; | |  | string phone3Param = "VsitPhone3: "+phone3; | |  | string platformIdParam = "PlatformID: 11"; | |  | curl\_global\_init(CURL\_GLOBAL\_ALL); | |  | readBuffer.clear(); | |  | curl = curl\_easy\_init(); | |  | if(curl) { | |  | struct curl\_slist \*chunk = NULL; | |  | chunk = curl\_slist\_append(chunk, emailParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, passParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, devIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, fNameParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, lNameParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, phone1Param.c\_str()); | |  | chunk = curl\_slist\_append(chunk, phone2Param.c\_str()); | |  | chunk = curl\_slist\_append(chunk, phone3Param.c\_str()); | |  | chunk = curl\_slist\_append(chunk, platformIdParam.c\_str()); | |  | curl\_easy\_setopt(curl, CURLOPT\_CUSTOMREQUEST, "POST"); | |  | curl\_easy\_setopt(curl, CURLOPT\_URL, "https://siv.voiceprintportal.com/sivservice/api/users"); | |  | res = curl\_easy\_setopt(curl, CURLOPT\_HTTPHEADER, chunk); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEFUNCTION, writeCallBack); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEDATA, &readBuffer); | |  | /\* Perform the request, res will get the return code \*/ | |  | res = curl\_easy\_perform(curl); | |  | /\* Check for errors \*/ | |  | if(res != CURLE\_OK) | |  | fprintf(stderr, "curl\_easy\_perform() failed: %s\n", | |  | curl\_easy\_strerror(res)); | |  | /\* always cleanup \*/ | |  | curl\_easy\_cleanup(curl); | |  | //cout<<"Response is "<<readBuffer<<" For Sure"<<endl; | |  | /\* free the custom headers \*/ | |  | curl\_slist\_free\_all(chunk); | |  | return readBuffer; | |  | } | |  | } | |  | string createUser(string email,string password,string firstName,string lastName) | |  | { | |  | return createUser(email, password, firstName, lastName, "","",""); | |  | } | |  | string deleteUser(string email,string password) | |  | { | |  | CURL \*curl; | |  | CURLcode res; | |  | string emailParam = "VsitEmail: "+email; | |  | string passParam = "VsitPassword: "+sha256(password); | |  | string devIdParam = "VsitDeveloperId: "+developerId; | |  | string platformIdParam = "PlatformID: 11"; | |  | curl\_global\_init(CURL\_GLOBAL\_ALL); | |  | readBuffer.clear(); | |  | curl = curl\_easy\_init(); | |  | if(curl) { | |  | struct curl\_slist \*chunk = NULL; | |  | chunk = curl\_slist\_append(chunk, emailParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, passParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, devIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, platformIdParam.c\_str()); | |  | curl\_easy\_setopt(curl, CURLOPT\_CUSTOMREQUEST, "DELETE"); | |  | curl\_easy\_setopt(curl, CURLOPT\_URL, "https://siv.voiceprintportal.com/sivservice/api/users"); | |  | res = curl\_easy\_setopt(curl, CURLOPT\_HTTPHEADER, chunk); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEFUNCTION, writeCallBack); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEDATA, &readBuffer); | |  | /\* Perform the request, res will get the return code \*/ | |  | res = curl\_easy\_perform(curl); | |  | /\* Check for errors \*/ | |  | if(res != CURLE\_OK) | |  | fprintf(stderr, "curl\_easy\_perform() failed: %s\n", | |  | curl\_easy\_strerror(res)); | |  | /\* always cleanup \*/ | |  | curl\_easy\_cleanup(curl); | |  | //cout<<"Response is "<<readBuffer<<" For Sure"<<endl; | |  | /\* free the custom headers \*/ | |  | curl\_slist\_free\_all(chunk); | |  | return readBuffer; | |  | } | |  | return ""; | |  | } | |  |  | |  | string createEnrollment(string email,string passwrd, string pathToEnrollmentWav, string contentLanguage){ | |  | CURL \*curl; | |  | CURLcode res; | |  | struct stat file\_info; | |  | //double speed\_upload, total\_time; | |  | FILE \*fd; | |  |  | |  | string emailParam = "VsitEmail: "+email; | |  | string passParam = "VsitPassword: "+sha256(passwrd); | |  | string devIdParam = "VsitDeveloperId: "+developerId; | |  | string contLangParam = "ContentLanguage: "+contentLanguage; | |  | string platformIdParam = "PlatformID: 11"; | |  | string cType = "Content-Type: audio/wav"; | |  | string accept = "Accept: application/json"; | |  | readBuffer.clear(); | |  |  | |  | fd = fopen(pathToEnrollmentWav.c\_str(), "rb"); /\* open file to upload \*/ | |  | if(!fd) { | |  |  | |  | return "Error"; /\* can't continue \*/ | |  | } | |  |  | |  | /\* to get the file size \*/ | |  | if(fstat(fileno(fd), &file\_info) != 0) { | |  |  | |  | return "Error"; /\* can't continue \*/ | |  | } | |  |  | |  | curl = curl\_easy\_init(); | |  | if(curl) { | |  | /\* upload to this place \*/ | |  | struct curl\_slist \*chunk = NULL; | |  | chunk = curl\_slist\_append(chunk, emailParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, passParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, devIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, contLangParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, platformIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, cType.c\_str()); | |  | chunk = curl\_slist\_append(chunk, accept.c\_str()); | |  | curl\_easy\_setopt(curl, CURLOPT\_CUSTOMREQUEST, "POST"); | |  | curl\_easy\_setopt(curl, CURLOPT\_URL, "https://siv.voiceprintportal.com/sivservice/api/enrollments"); | |  |  | |  | /\* tell it to "upload" to the URL \*/ | |  | curl\_easy\_setopt(curl, CURLOPT\_UPLOAD, 1L); | |  |  | |  | /\* set where to read from (on Windows you need to use READFUNCTION too) \*/ | |  | curl\_easy\_setopt(curl, CURLOPT\_READDATA, fd); | |  |  | |  | /\* and give the size of the upload (optional) \*/ | |  | //curl\_easy\_setopt(curl, CURLOPT\_INFILESIZE\_LARGE,(curl\_off\_t)file\_info.st\_size); | |  | res = curl\_easy\_setopt(curl, CURLOPT\_HTTPHEADER, chunk); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEFUNCTION, writeCallBack); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEDATA, &readBuffer); | |  | /\* enable verbose for easier tracing \*/ | |  | //curl\_easy\_setopt(curl, CURLOPT\_VERBOSE, 1L); | |  |  | |  | res = curl\_easy\_perform(curl); | |  | /\* Check for errors \*/ | |  | if(res != CURLE\_OK) { | |  | //fprintf(stderr, "curl\_easy\_perform() failed: %s\n",curl\_easy\_strerror(res)); | |  |  | |  | } | |  | else { | |  | /\* now extract transfer info \*/ | |  | //curl\_easy\_getinfo(curl, CURLINFO\_SPEED\_UPLOAD, &speed\_upload); | |  | //curl\_easy\_getinfo(curl, CURLINFO\_TOTAL\_TIME, &total\_time); | |  |  | |  | //fprintf(stderr, "Speed: %.3f bytes/sec during %.3f seconds\n",speed\_upload, total\_time); | |  |  | |  | } | |  | curl\_slist\_free\_all(chunk); | |  | /\* always cleanup \*/ | |  | curl\_easy\_cleanup(curl); | |  | return readBuffer; | |  | } | |  | return ""; | |  | } | |  |  | |  | string createEnrollment(string email,string passwrd, string pathToEnrollmentWav){ | |  |  | |  | return createEnrollment(email,passwrd, pathToEnrollmentWav, ""); | |  | } | |  |  | |  | string createEnrollmentByWavURL(string email,string passwrd, string urlToEnrollmentWav, string contentLanguage){ | |  | CURL \*curl; | |  | CURLcode res; | |  | //double speed\_upload, total\_time; | |  |  | |  | string emailParam = "VsitEmail: "+email; | |  | string passParam = "VsitPassword: "+sha256(passwrd); | |  | string devIdParam = "VsitDeveloperId: "+developerId; | |  | string wavURLParam = "VsitWavURL: "+urlToEnrollmentWav; | |  | string contLangParam = "ContentLanguage: "+contentLanguage; | |  | string platformIdParam = "PlatformID: 11"; | |  | string cType = "Content-Type: audio/wav"; | |  | string accept = "Accept: application/json"; | |  | readBuffer.clear(); | |  | curl = curl\_easy\_init(); | |  | if(curl) { | |  | /\* upload to this place \*/ | |  | struct curl\_slist \*chunk = NULL; | |  | chunk = curl\_slist\_append(chunk, emailParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, passParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, devIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, wavURLParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, contLangParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, platformIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, cType.c\_str()); | |  | chunk = curl\_slist\_append(chunk, accept.c\_str()); | |  | curl\_easy\_setopt(curl, CURLOPT\_CUSTOMREQUEST, "POST"); | |  | curl\_easy\_setopt(curl, CURLOPT\_URL, "https://siv.voiceprintportal.com/sivservice/api/enrollments/bywavurl"); | |  |  | |  |  | |  | /\* and give the size of the upload (optional) \*/ | |  | //curl\_easy\_setopt(curl, CURLOPT\_INFILESIZE\_LARGE,(curl\_off\_t)file\_info.st\_size); | |  | res = curl\_easy\_setopt(curl, CURLOPT\_HTTPHEADER, chunk); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEFUNCTION, writeCallBack); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEDATA, &readBuffer); | |  | /\* enable verbose for easier tracing \*/ | |  | //curl\_easy\_setopt(curl, CURLOPT\_VERBOSE, 1L); | |  |  | |  | res = curl\_easy\_perform(curl); | |  | /\* Check for errors \*/ | |  | if(res != CURLE\_OK) { | |  | //fprintf(stderr, "curl\_easy\_perform() failed: %s\n",curl\_easy\_strerror(res)); | |  |  | |  | } | |  | else { | |  | /\* now extract transfer info \*/ | |  | //curl\_easy\_getinfo(curl, CURLINFO\_SPEED\_UPLOAD, &speed\_upload); | |  | //curl\_easy\_getinfo(curl, CURLINFO\_TOTAL\_TIME, &total\_time); | |  |  | |  | //fprintf(stderr, "Speed: %.3f bytes/sec during %.3f seconds\n",speed\_upload, total\_time); | |  |  | |  | } | |  | curl\_slist\_free\_all(chunk); | |  | /\* always cleanup \*/ | |  | curl\_easy\_cleanup(curl); | |  | return readBuffer; | |  | } | |  | return ""; | |  | } | |  |  | |  | string createEnrollmentByWavURL(string email,string passwrd, string urlToEnrollmentWav){ | |  |  | |  | return createEnrollmentByWavURL(email,passwrd, urlToEnrollmentWav, ""); | |  | } | |  |  | |  | string getEnrollments(string email,string password) | |  | { | |  | CURL \*curl; | |  | CURLcode res; | |  | string emailParam = "VsitEmail: "+email; | |  | string passParam = "VsitPassword: "+sha256(password); | |  | string devIdParam = "VsitDeveloperId: "+developerId; | |  | string platformIdParam = "PlatformID: 11"; | |  | readBuffer.clear(); | |  | curl = curl\_easy\_init(); | |  | if(curl) { | |  | struct curl\_slist \*chunk = NULL; | |  | chunk = curl\_slist\_append(chunk, emailParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, passParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, devIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, platformIdParam.c\_str()); | |  | curl\_easy\_setopt(curl, CURLOPT\_URL, "https://siv.voiceprintportal.com/sivservice/api/enrollments"); | |  | res = curl\_easy\_setopt(curl, CURLOPT\_HTTPHEADER, chunk); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEFUNCTION, writeCallBack); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEDATA, &readBuffer); | |  | /\* Perform the request, res will get the return code \*/ | |  | res = curl\_easy\_perform(curl); | |  | /\* Check for errors \*/ | |  | if(res != CURLE\_OK) | |  | fprintf(stderr, "curl\_easy\_perform() failed: %s\n", | |  | curl\_easy\_strerror(res)); | |  | /\* always cleanup \*/ | |  | curl\_easy\_cleanup(curl); | |  | //cout<<"Response is "<<readBuffer<<" For Sure"<<endl; | |  | /\* free the custom headers \*/ | |  | curl\_slist\_free\_all(chunk); | |  | return readBuffer; | |  | } | |  | return "Error"; | |  | } | |  |  | |  | string deleteEnrollment(string email,string password,string enrollmentId) | |  | { | |  | CURL \*curl; | |  | CURLcode res; | |  | string finalURL = "https://siv.voiceprintportal.com/sivservice/api/enrollments/"+enrollmentId; | |  | string emailParam = "VsitEmail: "+email; | |  | string passParam = "VsitPassword: "+sha256(password); | |  | string devIdParam = "VsitDeveloperId: "+developerId; | |  | string platformIdParam = "PlatformID: 11"; | |  | curl\_global\_init(CURL\_GLOBAL\_ALL); | |  | readBuffer.clear(); | |  | curl = curl\_easy\_init(); | |  | if(curl) { | |  | struct curl\_slist \*chunk = NULL; | |  | chunk = curl\_slist\_append(chunk, emailParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, passParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, devIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, platformIdParam.c\_str()); | |  | curl\_easy\_setopt(curl, CURLOPT\_CUSTOMREQUEST, "DELETE"); | |  | curl\_easy\_setopt(curl, CURLOPT\_URL, finalURL.c\_str()); | |  | res = curl\_easy\_setopt(curl, CURLOPT\_HTTPHEADER, chunk); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEFUNCTION, writeCallBack); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEDATA, &readBuffer); | |  | /\* Perform the request, res will get the return code \*/ | |  | res = curl\_easy\_perform(curl); | |  | /\* Check for errors \*/ | |  | if(res != CURLE\_OK) | |  | fprintf(stderr, "curl\_easy\_perform() failed: %s\n", | |  | curl\_easy\_strerror(res)); | |  | /\* always cleanup \*/ | |  | curl\_easy\_cleanup(curl); | |  | //cout<<"Response is "<<readBuffer<<" For Sure"<<endl; | |  | /\* free the custom headers \*/ | |  | curl\_slist\_free\_all(chunk); | |  | return readBuffer; | |  | } | |  | return ""; | |  | } | |  |  | |  | string authentication(string email,string passwrd, string pathToAuthenticationWav, string confidence, string contentLanguage){ | |  | CURL \*curl; | |  | CURLcode res; | |  | struct stat file\_info; | |  | //double speed\_upload, total\_time; | |  | FILE \*fd; | |  |  | |  | string emailParam = "VsitEmail: "+email; | |  | string passParam = "VsitPassword: "+sha256(passwrd); | |  | string devIdParam = "VsitDeveloperId: "+developerId; | |  | string confidenceParam = "VsitConfidence: "+confidence; | |  | string contLangParam = "ContentLanguage: "+contentLanguage; | |  | string platformIdParam = "PlatformID: 11"; | |  | string cType = "Content-Type: audio/wav"; | |  | string accept = "Accept: application/json"; | |  | readBuffer.clear(); | |  |  | |  | fd = fopen(pathToAuthenticationWav.c\_str(), "rb"); /\* open file to upload \*/ | |  | if(!fd) { | |  |  | |  | return "Error"; /\* can't continue \*/ | |  | } | |  |  | |  | /\* to get the file size \*/ | |  | if(fstat(fileno(fd), &file\_info) != 0) { | |  |  | |  | return "Error"; /\* can't continue \*/ | |  | } | |  |  | |  | curl = curl\_easy\_init(); | |  | if(curl) { | |  | /\* upload to this place \*/ | |  | struct curl\_slist \*chunk = NULL; | |  | chunk = curl\_slist\_append(chunk, emailParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, passParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, devIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, confidenceParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, contLangParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, platformIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, cType.c\_str()); | |  | chunk = curl\_slist\_append(chunk, accept.c\_str()); | |  | curl\_easy\_setopt(curl, CURLOPT\_CUSTOMREQUEST, "POST"); | |  | curl\_easy\_setopt(curl, CURLOPT\_URL, "https://siv.voiceprintportal.com/sivservice/api/authentications"); | |  |  | |  | /\* tell it to "upload" to the URL \*/ | |  | curl\_easy\_setopt(curl, CURLOPT\_UPLOAD, 1L); | |  |  | |  | /\* set where to read from (on Windows you need to use READFUNCTION too) \*/ | |  | curl\_easy\_setopt(curl, CURLOPT\_READDATA, fd); | |  |  | |  | /\* and give the size of the upload (optional) \*/ | |  | //curl\_easy\_setopt(curl, CURLOPT\_INFILESIZE\_LARGE,(curl\_off\_t)file\_info.st\_size); | |  | res = curl\_easy\_setopt(curl, CURLOPT\_HTTPHEADER, chunk); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEFUNCTION, writeCallBack); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEDATA, &readBuffer); | |  | /\* enable verbose for easier tracing \*/ | |  | //curl\_easy\_setopt(curl, CURLOPT\_VERBOSE, 1L); | |  |  | |  | res = curl\_easy\_perform(curl); | |  | /\* Check for errors \*/ | |  | if(res != CURLE\_OK) { | |  | //fprintf(stderr, "curl\_easy\_perform() failed: %s\n",curl\_easy\_strerror(res)); | |  |  | |  | } | |  | else { | |  | /\* now extract transfer info \*/ | |  | //curl\_easy\_getinfo(curl, CURLINFO\_SPEED\_UPLOAD, &speed\_upload); | |  | //curl\_easy\_getinfo(curl, CURLINFO\_TOTAL\_TIME, &total\_time); | |  |  | |  | //fprintf(stderr, "Speed: %.3f bytes/sec during %.3f seconds\n",speed\_upload, total\_time); | |  |  | |  | } | |  | curl\_slist\_free\_all(chunk); | |  | /\* always cleanup \*/ | |  | curl\_easy\_cleanup(curl); | |  | return readBuffer; | |  | } | |  | return ""; | |  | } | |  |  | |  | string authentication(string email,string passwrd, string pathToAuthenticationWav, string confidence){ | |  |  | |  | return authentication(email,passwrd,pathToAuthenticationWav,confidence,""); | |  | } | |  |  | |  | string authenticationByWavURL(string email,string passwrd, string urlToAuthenticationWav, string confidence, string contentLanguage){ | |  | CURL \*curl; | |  | CURLcode res; | |  | //double speed\_upload, total\_time; | |  |  | |  | string emailParam = "VsitEmail: "+email; | |  | string passParam = "VsitPassword: "+sha256(passwrd); | |  | string devIdParam = "VsitDeveloperId: "+developerId; | |  | string confidenceParam = "VsitConfidence: "+confidence; | |  | string contLangParam = "ContentLanguage: "+contentLanguage; | |  | string wavURLParam = "VsitWavURL: "+urlToAuthenticationWav; | |  | string platformIdParam = "PlatformID: 11"; | |  | string cType = "Content-Type: audio/wav"; | |  | string accept = "Accept: application/json"; | |  | readBuffer.clear(); | |  | curl = curl\_easy\_init(); | |  | if(curl) { | |  | /\* upload to this place \*/ | |  | struct curl\_slist \*chunk = NULL; | |  | chunk = curl\_slist\_append(chunk, emailParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, passParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, devIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, confidenceParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, wavURLParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, contLangParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, platformIdParam.c\_str()); | |  | chunk = curl\_slist\_append(chunk, cType.c\_str()); | |  | chunk = curl\_slist\_append(chunk, accept.c\_str()); | |  | curl\_easy\_setopt(curl, CURLOPT\_CUSTOMREQUEST, "POST"); | |  | curl\_easy\_setopt(curl, CURLOPT\_URL, "https://siv.voiceprintportal.com/sivservice/api/authentications/bywavurl"); | |  |  | |  |  | |  | /\* and give the size of the upload (optional) \*/ | |  | //curl\_easy\_setopt(curl, CURLOPT\_INFILESIZE\_LARGE,(curl\_off\_t)file\_info.st\_size); | |  | res = curl\_easy\_setopt(curl, CURLOPT\_HTTPHEADER, chunk); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEFUNCTION, writeCallBack); | |  | curl\_easy\_setopt(curl, CURLOPT\_WRITEDATA, &readBuffer); | |  | /\* enable verbose for easier tracing \*/ | |  | //curl\_easy\_setopt(curl, CURLOPT\_VERBOSE, 1L); | |  |  | |  | res = curl\_easy\_perform(curl); | |  | /\* Check for errors \*/ | |  | if(res != CURLE\_OK) { | |  | //fprintf(stderr, "curl\_easy\_perform() failed: %s\n",curl\_easy\_strerror(res)); | |  |  | |  | } | |  | else { | |  | /\* now extract transfer info \*/ | |  | //curl\_easy\_getinfo(curl, CURLINFO\_SPEED\_UPLOAD, &speed\_upload); | |  | //curl\_easy\_getinfo(curl, CURLINFO\_TOTAL\_TIME, &total\_time); | |  |  | |  | //fprintf(stderr, "Speed: %.3f bytes/sec during %.3f seconds\n",speed\_upload, total\_time); | |  |  | |  | } | |  | curl\_slist\_free\_all(chunk); | |  | /\* always cleanup \*/ | |  | curl\_easy\_cleanup(curl); | |  | return readBuffer; | |  | } | |  | return ""; | |  | } | |  |  | |  | string authenticationByWavURL(string email,string passwrd, string urlToAuthenticationWav, string confidence){ | |  | return authenticationByWavURL(email,passwrd,urlToAuthenticationWav,confidence,""); | |  | } | |  | }; | |  |  | |  | #endif // \_VoiceIt\_ | |

**TESTING**

Testing created a handful of many different tests such as…

* successfully creating (enrolling) users
* failing to create due to incorrect information such as an email address
* getting users and failing to get users
* setting users and failing to set users
* authentication process successful and failures
* deleting users enrollment from the system

**Test File (test.cpp)**

<https://github.com/kcopper/Team7_Voice/blob/master/tests.cpp>

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | #include <assert.h> | |  | #include <cstdlib> | |  | #include <string> | |  | #include "VoiceIt.h" | |  |  | |  | using namespace std; | |  |  | |  | //Unit tests | |  |  | |  | void test\_Obj\_VoiceIt(){ | |  | cout<<"Testing VoiceIt object class."<<endl; | |  | try{ | |  | VoiceIt myVoiceIt("908e399927154baab125a5c2c9dc36af"); | |  | } | |  | catch(exception e){ | |  | cout<<"Failed to create VoiceIt Object."<<endl; | |  | } | |  | cout<<"Object creation succeeded."<<endl; | |  | VoiceIt myVoiceIt("908e399927154baab125a5c2c9dc36af"); | |  | cout<<"Testing object methods."<<endl; | |  | //Object method tests | |  | myVoiceIt.deleteUser("test@email.com", "testing"); | |  | //createUser | |  | string result = myVoiceIt.createUser("test@email.com", "testing", "Test", "McTester"); | |  | cout<<"Create User Test Success. "<<result<<endl; | |  | result = myVoiceIt.createUser("test", "invalid", "invalid", "invalid"); | |  | cout<<"Create User Test Invalid Email. "<<result<<endl; | |  | result = myVoiceIt.createUser("test@email.com", "testing", "Test", "McTester"); | |  | cout<<"Create User Test Already Exists. "<<result<<endl; | |  | result = myVoiceIt.createUser("test2@email.com", "testing2", "", "Burgundy"); | |  | cout<<"Create User Test No First. "<<result<<endl; | |  | result = myVoiceIt.createUser("test2@email.com", "testing2", "Ron", ""); | |  | cout<<"Create User Test No Last. "<<result<<endl; | |  | //getUser | |  | result = myVoiceIt.getUser("test@email.com", "testing"); | |  | cout<<"Get User Test Success. "<<result<<endl; | |  | result = myVoiceIt.getUser("test2@email.com", "testing"); | |  | cout<<"Get User Test User Not Found."<<result<<endl; | |  | result = myVoiceIt.getUser("email.com", "testing"); | |  | cout<<"Get User Test Invalid Email"<<result<<endl; | |  | result = myVoiceIt.getUser("test@email.com", "test"); | |  | cout<<"Get User Test Incorrect Password"<<result<<endl; | |  | //setUser | |  | result = myVoiceIt.setUser("test@email.com", "testing", "Test", "McTester"); | |  | cout<<"Set User Test Success. "<<result<<endl; | |  | result = myVoiceIt.setUser("email.com", "testing", "Test", "McTester"); | |  | cout<<"Set User Test Invalid Email. "<<result<<endl; | |  | result = myVoiceIt.setUser("test@email.com", "test", "Test", "McTester"); | |  | cout<<"Set User Test Incorrect Password. "<<result<<endl; | |  | result = myVoiceIt.setUser("test2@email.com", "tes", "Ron", "Burgundy"); | |  | cout<<"Set User Test User Not Found. "<<result<<endl; | |  | //createEnrollment | |  | result = myVoiceIt.createEnrollment("test@email.com", "testing", "TestPhraseWrong.wav", "en-US"); | |  | cout<<"Create Enrollment Test Wrong Phrase. "<<result<<endl; | |  | result = myVoiceIt.createEnrollment("email.com", "testing", "TestPhraseEnroll1.wav", "en-US"); | |  | cout<<"Create Enrollment Test Invalid Email. "<<result<<endl; | |  | result = myVoiceIt.createEnrollment("test@email.com", "test", "TestPhraseEnroll1.wav", "en-US"); | |  | cout<<"Create Enrollment Test Invalid Password. "<<result<<endl; | |  | result = myVoiceIt.createEnrollment("test2@email.com", "testing2", "TestPhraseEnroll1.wav", "en-US"); | |  | cout<<"Create Enrollment Test User Not Found. "<<result<<endl; | |  | cout<<"Authentication requires 3 valid enrollments"<<endl; | |  | result = myVoiceIt.createEnrollment("test@email.com", "testing", "TestPhraseEnroll1.wav", "en-US"); | |  | cout<<"Create Enrollment1 Test Success. "<<result<<endl; | |  | result = myVoiceIt.createEnrollment("test@email.com", "testing", "TestPhraseEnroll2.wav", "en-US"); | |  | cout<<"Create Enrollment2 Test Success. "<<result<<endl; | |  | result = myVoiceIt.createEnrollment("test@email.com", "testing", "TestPhraseEnroll3.wav", "en-US"); | |  | cout<<"Create Enrollment3 Test Success. "<<result<<endl; | |  | //getEnrollment | |  | result = myVoiceIt.getEnrollments("test@email.com", "testing"); | |  | cout<<"Get Enrollment Test Success. "<<result<<endl; | |  | size\_t start = result.find("[") + 1; | |  | size\_t end = 7; | |  | string num = result.substr(start, end); | |  | //authentication | |  | result = myVoiceIt.authentication("test@email.com", "testing", "TestPhraseAuth.wav", "90", "en-US"); | |  | cout<<"Authentication Test Success. Confidence 90 "<<result<<endl; | |  | result = myVoiceIt.authentication("test@email.com", "testing", "TestPhraseLow.wav", "90", "en-US"); | |  | cout<<"Authentication Test Fail, too low. Confidence 90"<<result<<endl; | |  | result = myVoiceIt.authentication("test@email.com", "testing", "TestPhraseHigh.wav", "90", "en-US"); | |  | cout<<"Authentication Test Fail, too high. Confidence 90"<<result<<endl; | |  | result = myVoiceIt.authentication("test@email.com", "testing", "TestPhraseWrong.wav", "90", "en-US"); | |  | cout<<"Authentication Test Fail, Wrong Phrase. Confidence 90"<<result<<endl; | |  | //deleteEnrollment | |  | result = myVoiceIt.deleteEnrollment("email.com", "testing", num); | |  | cout<<"Delete Enrollment Test Invalid Email. "<<result<<endl; | |  | result = myVoiceIt.deleteEnrollment("test@email.com", "tes", num); | |  | cout<<"Delete Enrollment Test Incorrect Password. "<<result<<endl; | |  | result = myVoiceIt.deleteEnrollment("test@email.com", "testing", "127"); | |  | cout<<"Delete Enrollment Test ID Doesn't Exist. "<<result<<endl; | |  | result = myVoiceIt.deleteEnrollment("test@email.com", "testing", num); | |  | cout<<"Delete Enrollment Test Success. "<<result<<endl; | |  | //deleteUser | |  | result = myVoiceIt.deleteUser("email.com", "testing"); | |  | cout<<"Delete User Invalid Email. "<<result<<endl; | |  | result = myVoiceIt.deleteUser("test@email.com", "test"); | |  | cout<<"Delete User Test Incorrect Password. "<<result<<endl; | |  | result = myVoiceIt.deleteUser("test@email.com", "testing"); | |  | cout<<"Delete User Test Success. "<<result<<endl; | |  | cout<<"Testing Complete."<<endl; | |  | } | |  |  | |  | int main(){ | |  | test\_Obj\_VoiceIt(); | |  | return 0; | |  | } | |

**CONCLUSION**

The Voice Authentication software created by Team 7 is what will be needed for this medical device to create a layer of security, ensuring that only authorized users (i.e. the patient’s doctor) will have access to the patient’s confidential medical data. The Voice Authentication software includes all of the client’s requirements, implements those requirements and properties effectively, has been tested and proven for accuracy, usability, and security, and can easily be integrated into the overall medical device. Team 7’s Voice Authentication software is the perfect solution for your medical device’s security and authentication, ensuring that your patient’s healthcare privacy is never sacrificed, and HIPPA is at all times adhered to when extracting data from this medical device.

**REFERENCES**

(2016, April 14). VoiceItTech – VoiceIt-cpp. Retrieved April 25, 2017, from https://github.com/voiceittech/voiceit-cpp/blob/master/VoiceIt.h

Schach, Stephen R. Object-Oriented Software Engineering. 1st ed. New York: McGraw-Hill Higher Education, 2008. Print.

Entire Group 7 Project Github Here: <https://github.com/kcopper/Team7_Voice>

**Copyright Information** © 2017, Team 7 (Voice). All Rights Reserved.