Prepared for:

Federal Communications Commission

CMS Alliance to Modernize Healthcare
Federally Funded Research and Development Center

Contract No. HHSM-5000-2012-000081

Task Order No. FCC15D0002

ACE Quill Release Documentation

Version 1.0

October 5, 2018

The views, opinions, and/or findings contained in this report are those of The MITRE Corporation and should not be construed as official government position, policy, or decision unless so designated by other documentation.

Approved for Public Release; Distribution Unlimited. Public Release Case Number 18-3244.

Record of Changes

Version	Date	Author / Owner	Description of Change
1.0	October 5, 2018	MITRE	Version 1.0 for Sponsor release

About the CMS Alliance to Modernize Healthcare

The Centers for Medicare & Medicaid Services (CMS) sponsors the CMS Alliance to Modernize Healthcare (CAMH), the first Federally Funded Research and Development Center (FFRDC) dedicated to strengthening our nation's healthcare system.

The CAMH FFRDC enables CMS, the Department of Health and Human Services (HHS), and other government entities to access unbiased research, advice, guidance, and analysis to solve complex business, policy, technology, and operational challenges in health mission areas. The FFRDC objectively analyzes long-term health system problems, addresses complex technical questions, and generates creative and cost-effective solutions in strategic areas such as quality of care, new payment models, and business transformation.

Formally established under Federal Acquisition Regulation (FAR) Part 35.017, FFRDCs meet special, long-term research and development needs integral to the mission of the sponsoring agency—work that existing in-house or commercial contractor resources cannot fulfill as effectively. FFRDCs operate in the public interest, free from conflicts of interest, and are managed and/or administered by not-for-profit organizations, universities, or industrial firms as separate operating units.

The CAMH FFRDC applies a combination of large-scale enterprise systems engineering and specialized health subject matter expertise to achieve the strategic objectives of CMS, HHS, and other government organizations charged with health-related missions. As a trusted, not-for-profit adviser, the CAMH FFRDC has access, beyond what is allowed in normal contractual relationships, to government and supplier data, including sensitive and proprietary data, and to employees and government facilities and equipment that support health missions.

CMS conducted a competitive acquisition in 2012 and awarded the CAMH FFRDC contract to The MITRE Corporation (MITRE). MITRE operates the CAMH FFRDC in partnership with CMS and HHS, and maintains a collaborative alliance of partners from nonprofits, academia, and industry. This alliance provides specialized expertise, health capabilities, and innovative solutions to transform delivery of the nation's healthcare services. Government organizations and other entities have ready access to this network of partners, including RAND Health, the Brookings Institution, and other leading healthcare organizations. This includes select qualified small and disadvantaged business.

The FFRDC is open to all CMS and HHS Operating Divisions and Staff Divisions. In addition, government entities outside of CMS and HHS can use the FFRDC with permission of CMS, CAMH's primary sponsor.

Executive Summary

The Federal Communication Commission (FCC) Telecommunications Relay Service (TRS) Center of Expertise (COE) Project promotes the Commission's goal to foster innovations that advance functionally equivalent telecommunication services for the Deaf and hard-of-hearing community. Toward that end, the project ensures that the Telecommunications Relay Service employs improved technology for persons who are Deaf, hard-of-hearing, Deaf-blind, and/or have speech disabilities. The FCC has embraced a research-based approach to achieve this goal by engaging the Centers for Medicare & Medicaid Services (CMS) Alliance to Modernize Healthcare (CAMH) Federally Funded Research and Development Center (FFRDC), operated by The MITRE Corporation (MITRE), to conduct independent engineering assessments that promote and demonstrate TRS's functional equivalence.

MITRE is independently assessing voice telephone services, video access services, and Internet Protocol (IP)-based caption telephone services (CTS); improvements to TRS efficiency; solutions for direct communication between people with communication disabilities and other telephone users; and the effectiveness and consumer response to current and future approaches for delivering TRS.

At the FCC's request, MITRE developed an open-source, test platform, mobile caption phone prototype to support device operability and usability assessments. The test platform prototype tests the speech-to-text (STT) application programming interfaces (API) available from Google Speech, IBM Watson, and Microsoft Bing Speech. The prototype enables direct calling by Deaf and hard-of-hearing individuals, automatically transcribing and captioning all audio from the called person for presentation to the Deaf and hard-of-hearing caller. (The test platform prototype can only make, but not receive, a call.) This document describes how to configure and use the mobile caption phone prototype for testing purposes.

The test prototype provides several benefits:

- **Improved Assessments** – The prototype improves device operability and usability assessments by using a standardized application to test multiple automatic speech recognition (ASR) and STT technologies on critical attributes of accuracy and latency.
- **Reduced Variation –** The prototype reduces the variability that arises from changing devices and applications. It promotes a user perception testing comparison and automated device testing comparison of ASR and STT technologies and provides vendors of advancing technologies a better understanding of the minimum performance required to compare with the currently available IP CTS framework.
- **Improved Communications** – ASR technology provides captioned transcripts to the user and may improve efficiency, effectiveness, and productivity over traditional IP CTS.
- **Cost Savings –** ASR-based captioning capabilities reduce the time to construct, perform, and collect data for test analysis. The capability to augment caption telephone conversations may reduce the total cost of caption calls while improving the efficiency and effectiveness of these calls, but no comparisons or cost analysis have been performed. This prototype may also save development and operational costs for other researchers and developers who produce an automated IP CTS.

Because the current prototype is a laboratory and research testing tool limited to originating but not receiving a call. It is unsuitable for general-purpose consumer use. Subsection 1.4.1 describes other applicable limitations.

As part of this effort, MITRE developed and documented the capabilities and features for the test platform prototype and associated server-side components. Detailed configuration and source code files will be made available for download and reproduction to improve solutions to support the community.

Table of Contents

List of Figures

List of Tables

1. Introduction

The Federal Communications Commission (FCC) Telecommunications Relay Service (TRS) Center of Expertise (COE) Project promotes the Commission's goal to foster innovations that advance functionally equivalent telecommunications. Toward that end, the project ensures that the Telecommunications Relay Service employs improved technology for persons who are Deaf, hard-of-hearing, Deaf-blind, and/or have speech disabilities.

1.1 Background

The FCC has embraced a research-based approach to achieve this goal by engaging the Centers for Medicare & Medicaid Services (CMS) Alliance to Modernize Healthcare (CAMH) Federally Funded Research and Development Center (FFRDC), operated by The MITRE Corporation (MITRE), to conduct independent engineering assessments that promote and demonstrate TRS' functional equivalence.

MITRE is independently assessing voice telephone services, video access services, and Internet Protocol (IP)-based caption telephony; improvements to TRS efficiency; solutions for direct communication between people with communication disabilities and other telephone users; and the effectiveness, efficiency, and consumer response to current and future approaches for delivering TRS.

At the FCC's request, MITRE has developed an open-source test platform prototype for a mobile caption phone to support device operability and usability assessments of IP Caption Telephone Services (IP CTS) devices. The test platform prototype enables direct calling by Deaf and hard-of-hearing individuals and automatically transcribes and captions audio from the callee and presents it to the caller. (At present, the test platform prototype can only make, but not receive, a call. Subsection 1.4.1 addresses the known limitations of the prototype.)

This test platform prototype uses the current, market-leading speech-to-text (STT) application program interfaces (API) available from Google Speech, IBM Watson, and Microsoft Bing Speech. Additional STT APIs may be examined in future testing.

As part of this effort, MITRE developed and documented the requirements and features for the test platform prototype of an open-source, mobile caption phone. MITRE also configured, tested, and validated the prototype. Detailed configuration and source code files are available for download to improve existing solutions and encourage further research by outside institutions.

This document details the requirements for developing the application, including the architecture, functional requirements, and recommendations for design, construction of a common user interface, and how to configure and use the prototype for testing.

1.2 Internet Protocol Caption Telephone Service

Deaf, hard-of-hearing, Deaf-blind, and or speech-disabled people use TRS to communicate with hearing people. IP CTS is a TRS designed for an individual who can speak, but who has difficulty hearing over the telephone. In most cases, an individual can use a telephone and an IP-enabled device to listen to the other party and simultaneously read captions of the other party's words. In most cases, IP CTS involves the use of third-party communication assistants (CA).

These CAs listen during the call and repeat, or “re-voice,” everything the other party says into a STT transcription program, or other device, to produce captions for the Deaf or hard-of-hearing user. With the popularity of mobile devices, IP CTS service providers now offer mobile applications that provide re-voicing services for the Deaf and hard-of-hearing community.

1.3 Prototype Overview

The FCC’s National Test Laboratory (NTL) sponsorship includes ongoing evaluation and testing of IP CTS operability. The NTL has created an open-source test platform prototype of a caption telephone to facilitate and upgrade testing methodologies for improving technologies. Known as “Accessible Communications for Everyone (ACE) Quill,” the ACE Quill prototype will improve operability and usability assessments by using a standardized application to test multiple Automatic Speech Recognition (ASR) technologies. The ACE Quill test platform prototype now supports two primary use cases:

1. **Study participants with hearing loss**
    a. Study participants can use the prototype to make phone calls
    b. The prototype transcribes the audio signal into text in the form of captions displayed to the study participants (call audio is also returned)
2. **Captioning research in accuracy and latency**
    a. The prototype allows researchers to select the ASR/STT technology used to caption text.
    b. Researchers can control the application of latency filters.
    c. The prototype generates reports containing call data records and raw transcripts of the call.

The ACE Quill prototype will reduce the variability that arises from changing devices and applications. This will facilitate a controlled comparison of ASR technologies and ensure a better understanding of the minimum performance required for advancing technologies to augment the currently available IP CTS framework.

1.4 Conceptual System Overview

The ACE Quill prototype consists of several software components. The major components in the architecture are:

- **ACE Quill Android application –** Places the phone call and displays the transcript text to the user
- **ACE Quill Server –** Interfaces with the STT engines
- **Asterisk Public Branch Exchange (PBX) –** Virtual phone switch that routes calls
- **STT Engines –** Generate transcript text from the audio of the callee side (the called person) of the phone call

- **Research Portal –** Web-based application that displays call data for researchers
- **MySQL –** Open-source relational database used to store the research data and user login credentials

Figure 1 presents a notional view of the architecture and shows the type of messaging between the components. For example, the dashed lines indicate Session Initiation Protocol (SIP) messaging and the solid lines indicate a Transmission Control Protocol (TCP) connection using a well-defined API. The numbered circles in the figure represent the steps listed in the next paragraph.

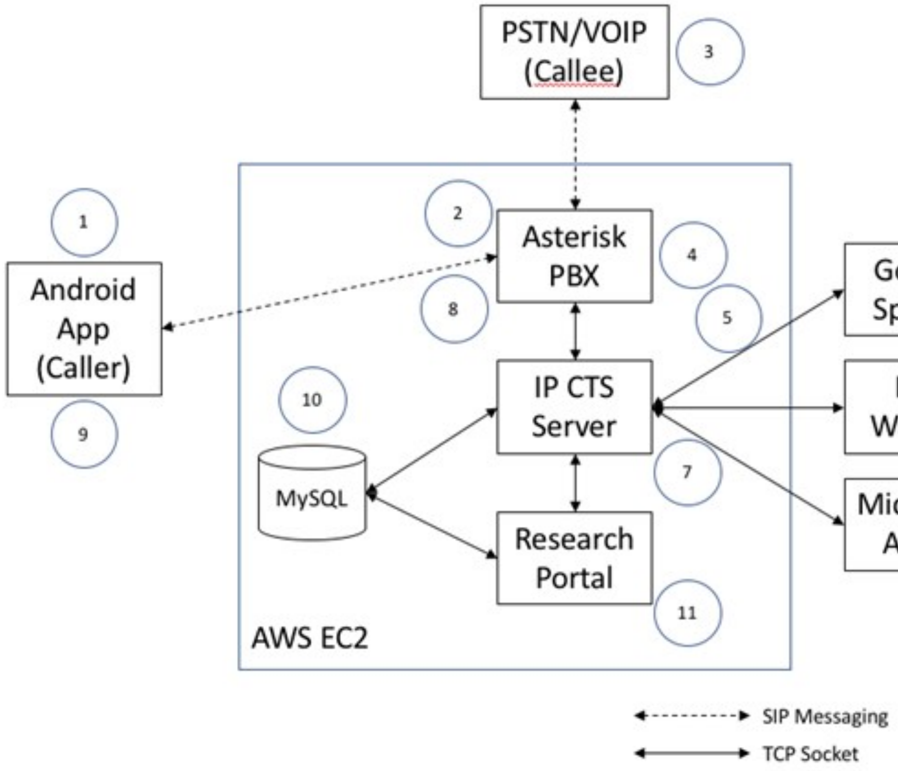


Figure 1. IP CTS Notional Architecture

The ACE Quill prototype shown in Figure 1 operates as follows:

- **Step 1:** The caller opens the Android app, enters a destination phone number, and presses the dial button to initiate the call. The Android app uses SIP messaging to communicate with the Asterisk PBX.
- **Step 2:** Asterisk PBX routes the call to the destination phone number. The Android app can only place outbound calls; it cannot receive a call.
- **Step 3:** The callee answers the incoming call. Any phone type, whether mobile or public switched telephone network (PSTN), can be used as the destination for the call.
- **Step 4:** As soon as the call is answered, the Asterisk PBX begins recording the callee leg of the call as a WAV file.

- **Step 5:** The ACE Quill server reads the WAV file and streams the audio to the configured STT engine via IP.
- **Step 6:** The selected STT engine transcribes the incoming audio stream and generates a corresponding JavaScript Object Notation (JSON) object containing the transcribed text.
- **Step 7:** The transcribed text is sent from the STT engine to the IP CTS server.
- **Step 8:** The ACE Quill server sends the transcribed text to the Android app through the Asterisk PBX using SIP messages.
- **Step 9:** The Android app displays the transcribed text.
- **Step 10:** After the call is completed, data about the call (e.g., user identifier, device, start time, stop time, and file names) is stored in the MySQL database.
  - The user name here can be an actual name (e.g., George Washington) or a shortened name (e.g., gwashington).
  - User names are created in the administration section of the portal.
- **Step 11:** The Research Portal can be used to view call data, replay audio files, display transcripts, and download reports.

1.4.1 Limitations

The ACE Quill Prototype is intended solely as a tool for testing captions and does not include all features of a full-function telephone. Successful adaption of the ACE Quill prototype for commercial use would necessitate some of the following features:

- **iOS support** – At present, the test platform prototype is only supported on the Android operating system, which would limit the potential customer base.
- **Inbound call capability** – The prototype only supports outbound calls (calls placed from the prototype).
- **Bi-directional transcription** – The prototype only supports transcription of the inbound audio, like current IP CTS services. Bi-directional transcription, where both sides of the conversation are displayed on the device, may be desirable in a commercial application or other research applications.
- **Typical phone-related features –** Call history, contact lists, and voicemail are not related to transcription testing, and therefore, are not supported in the prototype.
- **911 capabilities –** The prototype does not support any 911 or e-911 services.

Note: A usability team should be involved in user research on all proposed changes and testing.

1.5 Open-source Development to Promote Community Involvement

Open-source promotes universal access via a free license to a product’s design/blueprint and universal redistribution of that design/blueprint, including subsequent improvements to it. The open-source model employs a decentralized model of production. A main principle of open-source software development is peer production: products such as source code, “blueprints,” and documentation, are available to the public at no cost.

The FCC encourages government agencies, educational institutions, and others seeking to enhance the lives of citizens who are Deaf, HoH, Deaf-blind, and/or have speech disabilities to adopt and improve on the existing code base. Improving the code base could involve adding new capabilities, improving the workflow, and introducing new technologies to the open-source ACE Quill platform.

2. ACE Quill Operation

2.1 Launching the ACE Quill Android App

The ACE Quill app is not available in the Google Play store. Subsection 3.1.4 presents instructions for installing the app on an Android phone.

Once the app is installed, the researcher starts the client. Locate the ACE Quill icon on the Android desktop and click on the icon. Figure 2 shows a screenshot of an Android desktop with the ACE Quill icon highlighted by the dotted-line box. This screenshot was obtained from a Google Pixel and may vary slightly on other Android devices.

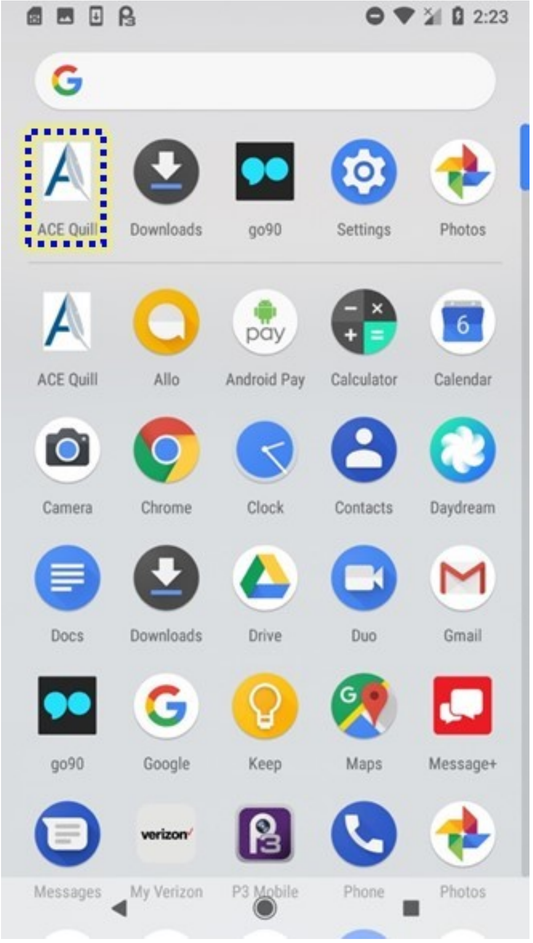
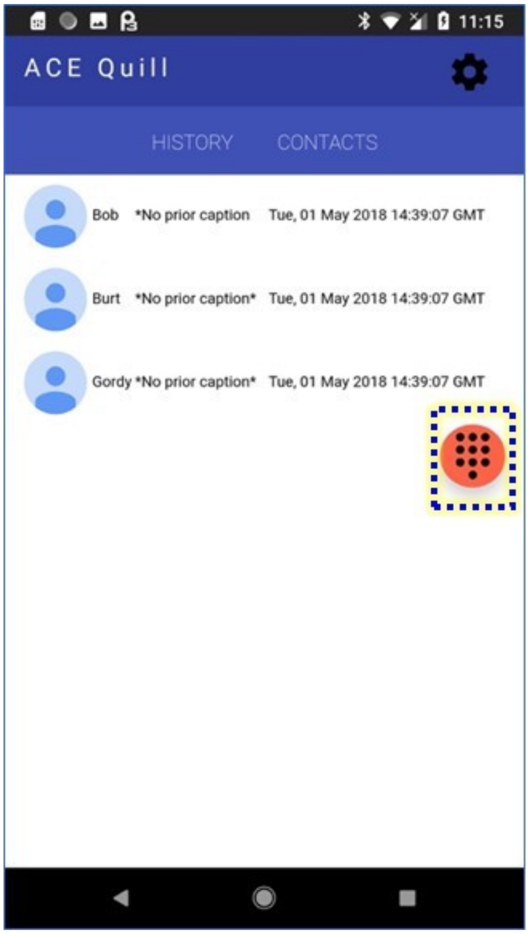


Figure 2. Screenshot of Typical Android Desktop

Click on the ACE Quill icon to launch the app. Figure 3 shows the opening screen for the ACE Quill app. **Note:** If the app is not configured, the user is directed to the Settings page on startup. For this example, however, it is assumed that the app is configured. Please refer to subsection 3.1.5 for a discussion of App configuration.

2.2 Initiating a Call

Select the read dial pad icon (highlighted by the dotted-line box) to open the dial pad.



Bob *No prior caption Tue, 01 May 2018 14:39:07 GMT

Burt *No prior caption* Tue, 01 May 2018 14:39:07 GMT

Gordy *No prior caption* Tue, 01 May 2018 14:39:07 GMT

Figure 3. Screenshot of ACE Quill Opening Screen

Once the dial pad is open, operate the app like any mobile telephone. Figure 4 shows the dial pad for the ACE Quill prototype. To place a call, enter the phone number using the dial pad. The app will display the entered phone number in the grey bar above the dial pad. Press the “Delete” icon in this bar to delete and adjust numbers, if needed. Press the green telephone icon (highlighted by the dotted-line box) located under the dial pad to initiate the call.

In Figure 4, the History and Contacts buttons are placeholders for future implementation and do not function. Refer to subsection 1.4.1 for limitations of the current prototype application.

The gear icon (located in the upper right) will open the Settings page. Any changes to the Settings page are saved to the app.

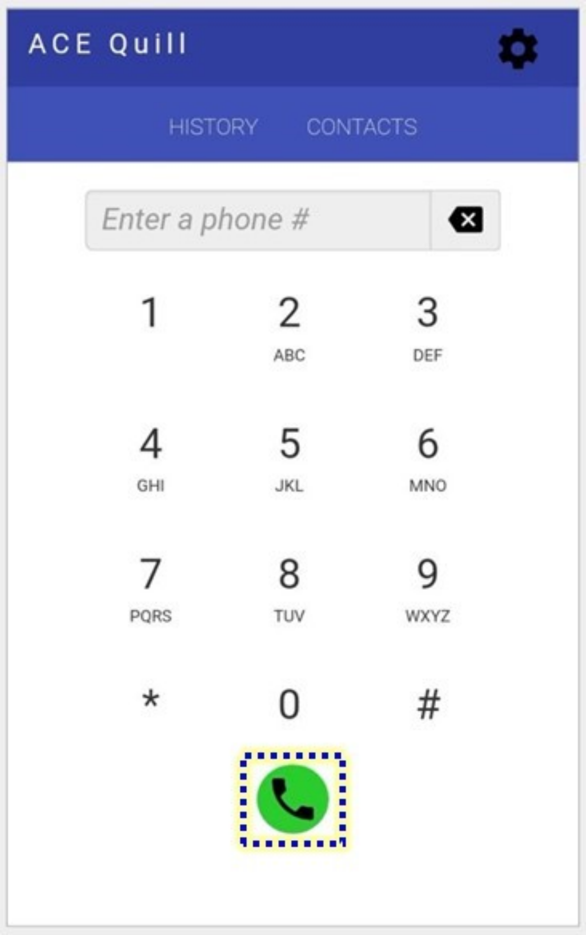


Figure 4. Screenshot of Application Dial Pad

Entering a phone number and pressing the dial button (highlighted by the dotted-line box) will initiate a phone call. During a call, the app will display a screen with text boxes showing incoming transcribed data. Once connected, the transcript of the incoming callee side of the conversation will be displayed on the Android app for the duration of the phone call. After the call ends, the user will be returned to the dial pad. Figure 5 provides an example of a generated transcript from a scripted call about ordering a pizza.

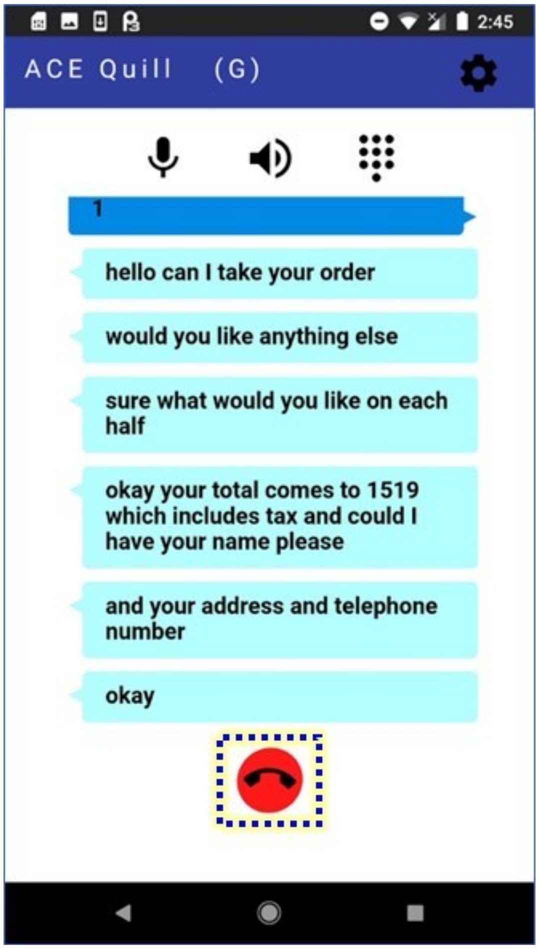
The transcribed data displayed in light blue text boxes originates from the left side of the screen. This is the callee data. Because the app orientation is locked in portrait mode, the display will not change if the phone is rotated.

The following icons appear during a call:

- The dial pad icon on the upper right of the screen will toggle between hiding and displaying the dial pad. This feature allows more screen space to display transcribed text, when needed.
- The speaker icon will mute audio output during a call.

The microphone icon is a placeholder for a future function. In the example screenshot on Figure 5, a "G" appears at the top center of the screen. This indicates that the Google STT engine is active. If the Microsoft STT engine were used, the app would display an "A". Similarly, the app would display a "W" if the IBM Watson STT engine were used.

To end the call, press the red hang-up button, highlighted by the dotted-line box.



hello can I take your order

would you like anything else

sure what would you like on each half

okay your total comes to 1519 which includes tax and could I have your name please

and your address and telephone number

okay

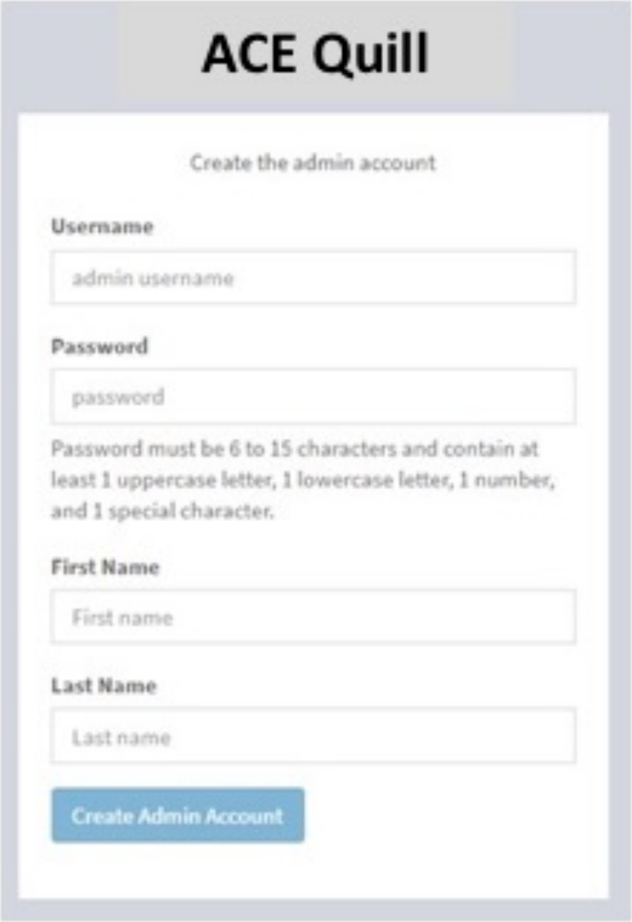
Figure 5. Screenshot of Sample Transcript Result

2.3 Research Portal

The Research Portal is a web-based application that allows the researcher to configure the system and display the data collected about the call. It consists of three main sections: Settings, Reports, and Administrator. This portal provides the research team with the capabilities to configure the Android phones, enter credentials for each ASR engine, review call records, add/remove users, and view server status.

The ACE Quill server hosts the Research Portal; therefore, the server must be running to access the page. Once installed on a local server, the URL for the research portal will be similar to: https:/<your /host name>:<your config port>/admin.

On an initial login, the user database is empty. The user will be presented with a screen to create the administration account as shown in Figure 6.



ACE Quill

Create the admin account

Username

admin username

Password

password

Password must be 6 to 15 characters and contain at least 1 uppercase letter, 1 lowercase letter, 1 number, and 1 special character.

First Name

First name

Last Name

Last name

Figure 6. Screenshot of First Login Prompt

After the administrative account is created, the user will see the login prompt shown in Figure 7 on any subsequent login attempts.

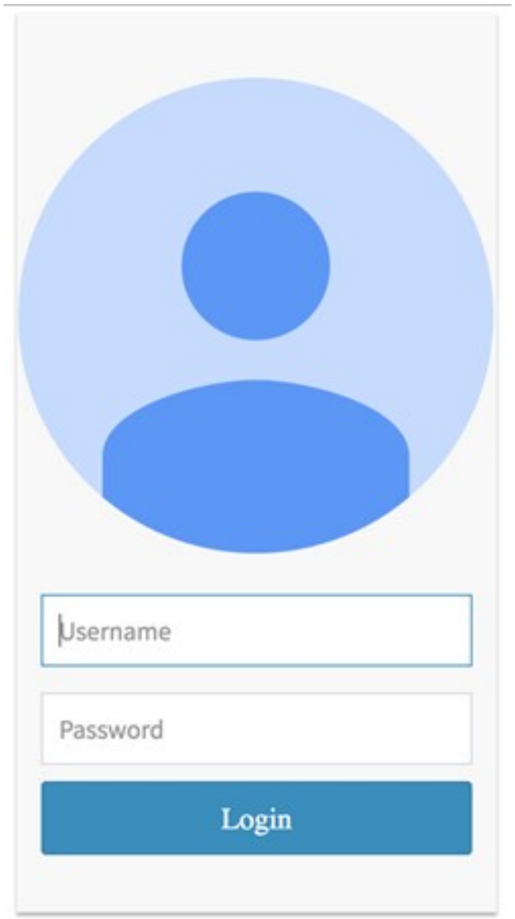


Figure 7. Screenshot of Login Prompt

To enter the Research Portal, provide the administration username and password and press Login. To navigate inside the Research Portal, use the links on the left side of the page. The following subsections address each of the functions in the Research Portal.

2.3.1 Research Portal Settings

The Settings section of the ACE Quill Research Portal allows the user to configure user accounts and specify STT engine credentials. The portal offers settings for the functions of Configuration and SST Engines, as discussed in the following subsections.

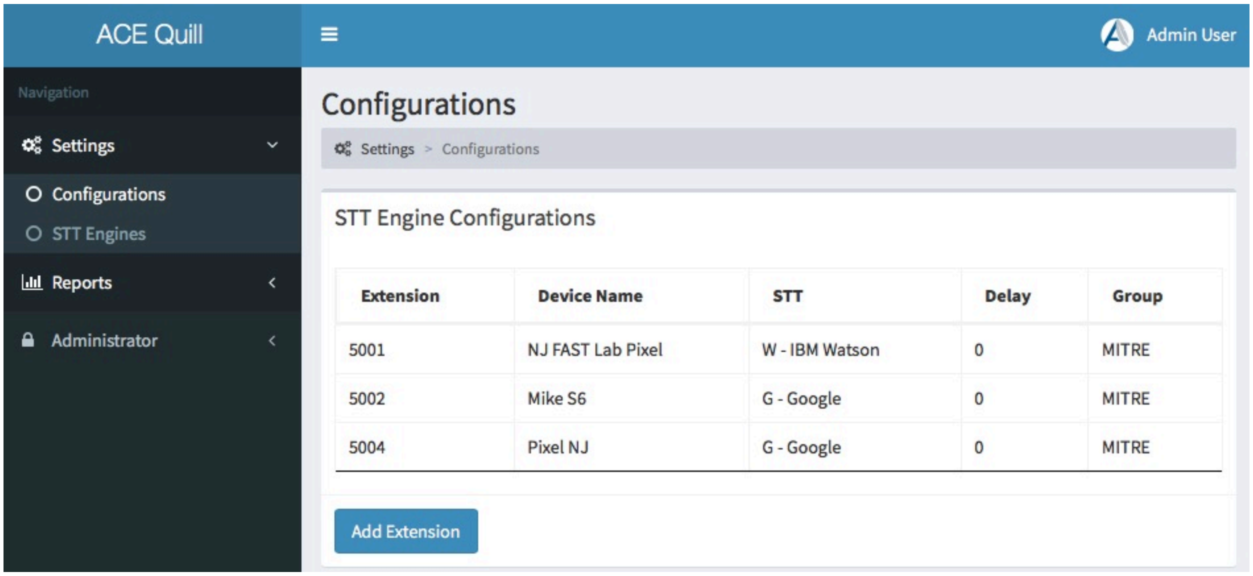
2.3.1.1 Configurations Page

The Configurations page allows the user to uniquely identify the devices that will be used with the ACE Quill prototype. The app supports setting parameters for:

- **Extension** – A unique number assigned to each Android device. Currently, extensions 5001 through 5004 are valid; additional extensions can be added by extending the Asterisk sip.conf file.
- **Device name** – A name assigned to the Android phone. The name could be based on the user’s name, location, etc.

- **Default STT Engine –** Any of the supported STT engines (Google Speech, Microsoft Bing Speech, and IBM Watson) can be used by any Android phone registered with the system; however, each Android phone can be assigned a different default STT engine.
- **Delay –** An extra delay can be added to the return of the transcript. The delay is configurable from 1 to 25 seconds in full-second values and can help a research team investigate the effect of delay on the user experience.

Figure 8 shows the default settings for each extension. In this example, extension 5001 will use the IBM Watson STT engine and extensions 5002 and 5004 will use the Google STT engine. A group is also assigned to each extension, which protects data validity by limiting the visibility of the research data by separate research teams; accordingly, a user may only see calls made by a member of their group. Research groups prevent unintended data sharing, especially if different research groups or even organizations are using the same server.



STT Engine Configurations

Extension	Device Name	STT	Delay	Group
5001	NJ FAST Lab Pixel	W - IBM Watson	0	MITRE
5002	Mike S6	G - Google	0	MITRE
5004	Pixel NJ	G - Google	0	MITRE

Figure 8. Screenshot of Default STT Engine Configuration

2.3.1.2 Speech-to-Text Engines Page

The STT Engines page contains three tabs—one for each of the supported STT engines (Microsoft Bing Speech, Google Speech, and IBM Watson). Once accounts are created with each of STT engine providers, credentials can be downloaded in a JSON file. The account information from the JSON files can be entered in each respective tab. Figure 9 shows the configuration page for the STT engines. **Note:** Each STT engine has different required fields for account credentials.

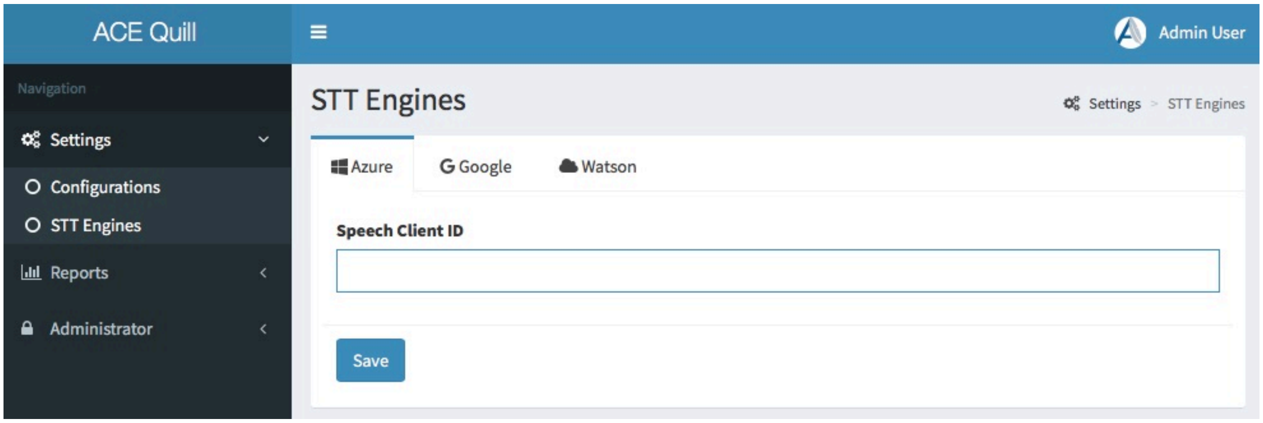


Figure 9. Screenshot of STT Engine Configuration

The links to create the STT engine accounts are:

- Google Speech
  - https://cloud.google.com
- IBM Watson
  - https://www.ibm.com/watson/services/speech-to-text/
- Microsoft Bing Speech
  - https://account.azure.com/Home/Index

2.3.2 Reports

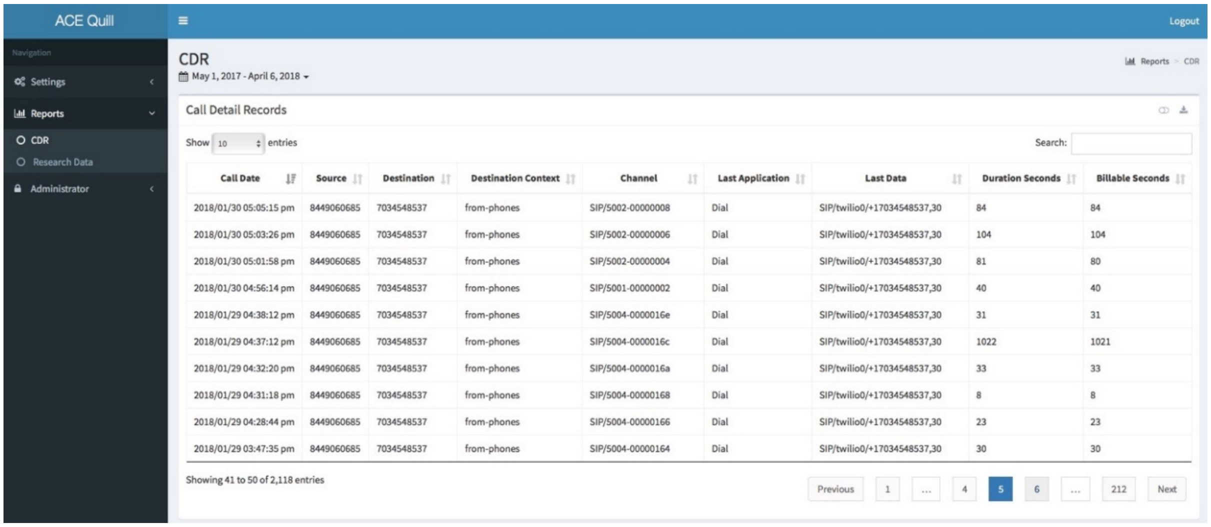
The Reports section contains two reports—Call Detail Records (CDR) and Research Data Page—that can be displayed and downloaded. The following subsections describe each report.

2.3.2.1 Call Detail Records

The Call Detail Record report is an Asterisk-generated report that contains one row per phone call. The following fields are included in this report:

- **Call date** – The time and date of a call.
- **Source** – The extension of the device that initiated the call (caller).
- **Destination** – The destination phone number (callee).
- **Destination context** – The destination context for the call.
- **Channel** – Unique identifier assigner to the callee leg of the call.
- **Last application** – The last dialplan application executed during the call.
- **Last data** – The data arguments passed to the last application.
- **Duration seconds** – The time between the start and end time of the call.
- **Billable seconds** – The time between the answer and end time of the call.

The CDR table contains sortable columns and date filtering and supports download as a comma-separated values (CSV) file. Figure 10 shows an example of the CDR table.



CDR

May 1, 2017 - April 6, 2018

Call Detail Records

Show 10 entries Search:

Call Date	Source	Destination	Destination Context	Channel	Last Application	Last Data	Duration Seconds	Billable Seconds
2018/01/30 05:05:15 pm	8449060685	7034548537	from-phones	SIP/5002-00000008	Dial	SIP/twilio0/+17034548537,30	84	84
2018/01/30 05:03:26 pm	8449060685	7034548537	from-phones	SIP/5002-00000006	Dial	SIP/twilio0/+17034548537,30	104	104
2018/01/30 05:01:58 pm	8449060685	7034548537	from-phones	SIP/5002-00000004	Dial	SIP/twilio0/+17034548537,30	81	80
2018/01/30 04:56:14 pm	8449060685	7034548537	from-phones	SIP/5001-00000002	Dial	SIP/twilio0/+17034548537,30	40	40
2018/01/29 04:38:12 pm	8449060685	7034548537	from-phones	SIP/5004-0000016e	Dial	SIP/twilio0/+17034548537,30	31	31
2018/01/29 04:37:12 pm	8449060685	7034548537	from-phones	SIP/5004-0000016c	Dial	SIP/twilio0/+17034548537,30	1022	1021
2018/01/29 04:32:20 pm	8449060685	7034548537	from-phones	SIP/5004-0000016a	Dial	SIP/twilio0/+17034548537,30	33	33
2018/01/29 04:31:18 pm	8449060685	7034548537	from-phones	SIP/5004-00000168	Dial	SIP/twilio0/+17034548537,30	8	8
2018/01/29 04:28:44 pm	8449060685	7034548537	from-phones	SIP/5004-00000166	Dial	SIP/twilio0/+17034548537,30	23	23
2018/01/29 03:47:35 pm	8449060685	7034548537	from-phones	SIP/5004-00000164	Dial	SIP/twilio0/+17034548537,30	30	30

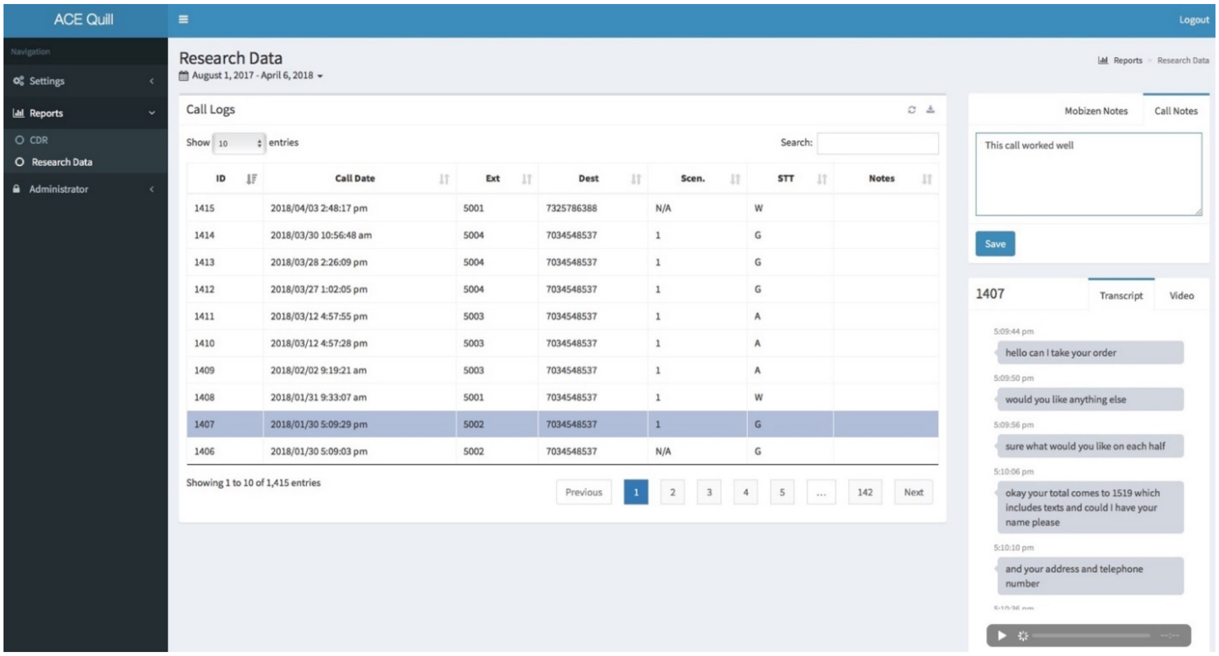
Showing 41 to 50 of 2,118 entries

Figure 10. Screenshot of Call Detail Records Table

The CDR report contains valuable information for developers to resolve call-related issues that need debugging. The web-based table displays a subset of the records in the database table. An explanation of the default CDR fields can be found here: http://www.asteriskdocs.org/en/3rd_Edition/asterisk-book-html-chunk/asterisk-SysAdmin-SECT-1.html.

2.3.2.2 Research Data Page

The Research Data page displays the data collected during each call by the ACE Quill application. Figure 11 shows an example of the Research Data page.



Research Data

August 1, 2017 - April 6, 2018

Call Logs

Show 10 entries Search:

ID	Call Date	Ext	Dest	Scen.	STT	Notes
1415	2018/04/03 2:48:17 pm	5001	7325786388	N/A	W	
1414	2018/03/30 10:56:48 am	5004	7034548537	1	G	
1413	2018/03/28 2:26:09 pm	5004	7034548537	1	G	
1412	2018/03/27 1:02:05 pm	5004	7034548537	1	G	
1411	2018/03/12 4:57:55 pm	5003	7034548537	1	A	
1410	2018/03/12 4:57:28 pm	5003	7034548537	1	A	
1409	2018/02/02 9:19:21 am	5003	7034548537	1	A	
1408	2018/01/31 9:33:07 am	5001	7034548537	1	W	
1407	2018/01/30 5:09:29 pm	5002	7034548537	1	G	
1406	2018/01/30 5:09:03 pm	5002	7034548537	N/A	G	

Showing 1 to 10 of 1,415 entries

This call worked well

1407

5:09:44 pm
hello can I take your order

5:09:50 pm
would you like anything else

5:09:56 pm
sure what would you like on each half

5:10:06 pm
okay your total comes to 1519 which includes texts and could I have your name please

5:10:10 pm
and your address and telephone number

Figure 11. Screenshot of Research Data Page

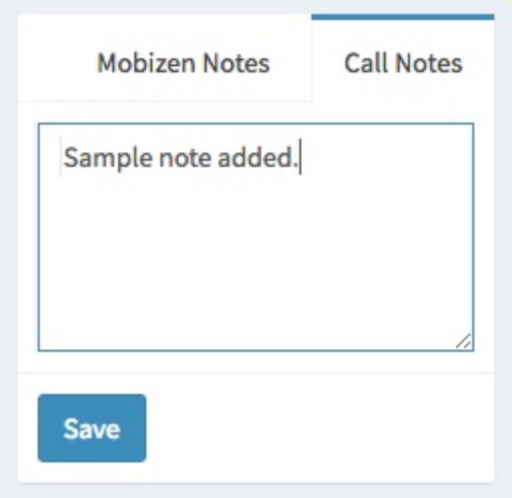
There are three main areas on the Research Data page:

- Call Logs (large table in center of screen)
- Mobizen Notes/Call Notes (upper right of screen)
- Transcript/Video (lower right of screen)

The Call Logs table contains a record for each call that was made by the ACE Quill Android app. The following columns are displayed for each call:

- **ID –** A unique identifier assigned to the call.
- **Call Date –** Timestamp identifying when the call was made.
- **Ext –** Identifies the extension (Android phone) that made the call.
- **Dest –** Shows the destination phone number (callee).
- **Scenario –** Shows the destination phone extension if calling an automated scenario server.
- **STT –** Identifies the STT engine used to transcribe the call.
- **Notes –** Shows any notes that the researcher added to this call. For example, transcripts were delayed, text was missing, etc.

Clicking on a row (call) in the Call Logs table auto-populates the Call Notes section and the Transcript/Video section. Figure 12 shows the Mobizen and Call Notes input section.

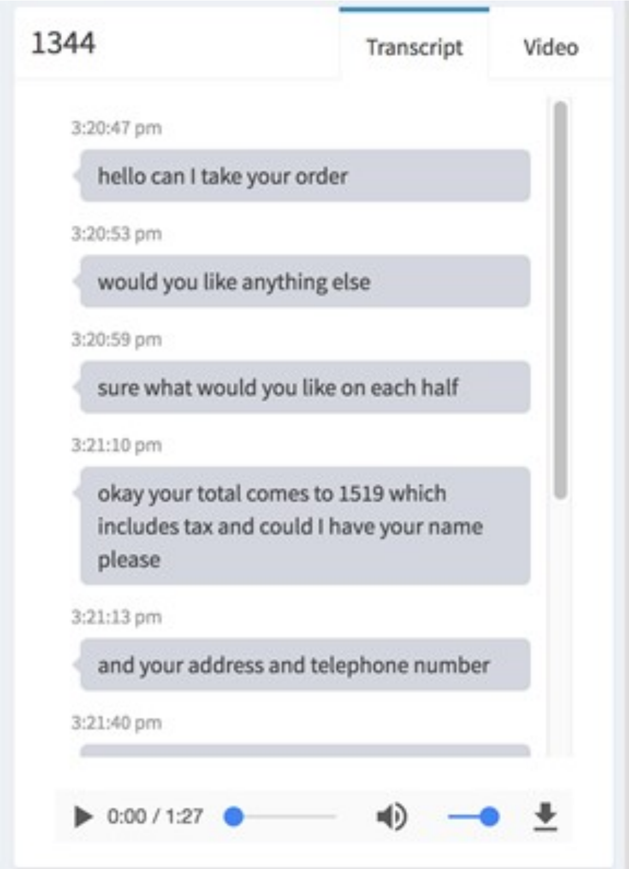


Sample note added.

Figure 12. Screenshot of Mobizen and Notes Input

The Call Notes can be used to log any notes about the call for review later. A researcher may want to log information about a test subject, an error, etc. Mobizen is an Android app for recording the screen during a call. Researchers wanted an area to log any information related to the call recording.

Both Mobizen and Call Notes can be added or edited while a row is highlighted, and the transcribed text for a call is displayed on the transcript tab. Sometimes, the researchers may record a video of the research participant making a call. The Transcript/Video section supports uploading a video recording of the call. The recorded audio call can also be replayed on the Transcript tab. Figure 13 shows an example of a transcript for call number 1344.



3:20:47 pm
hello can I take your order

3:20:53 pm
would you like anything else

3:20:59 pm
sure what would you like on each half

3:21:10 pm
okay your total comes to 1519 which includes tax and could I have your name please

3:21:13 pm
and your address and telephone number

3:21:40 pm

Figure 13. Screenshot of Sample Transcript

In the top right of the Call Logs table, there is a button to download the data in the Research Portal as a CSV file. Table 1 describes the report fields and offers sample data.

Table 1. Format of the Research Data Download

Field Name	Description	Sample Data
Id	Unique ID number to identify a call	135
call_start	Start time of the phone call in GMT	2017-08-22T19:13:05.000Z
device_id	Unique ID to identify the device that initiated the phone call	c92e33c6770d0629
extension	Extension number assigned to the phone	5001
dest_phone_number	Destination phone number	1234567890
call_duration	Phone call duration in seconds	38
scenario_number	The scenario selected during an interactive scenario	2

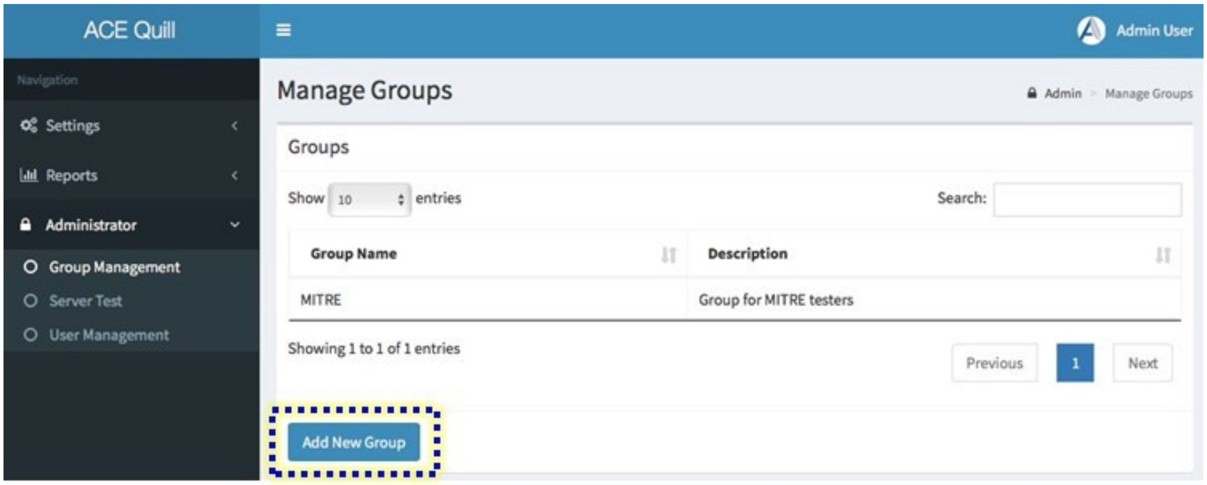
Field Name	Description	Sample Data
stt_engine	The name of the STT engine	AZURE (Microsoft Bing Speech), GOOGLE, WATSON
added_delay	Added delay (in seconds) added to the transcription	5
transcription_file	Link to download the call transcript as a text file	https://...
audio_file	Link to download the call transcript as a WAV file	https:/…
transcription_filename	Name of the call transcript text file	08-22-2017_19-13-05_SIP-twilio0-000001a3.txt
audio_filename	Name of the call transcript WAV file	08-22-2017_19-13-05_SIP-twilio0-000001a3-pstn-in.wav16
notes	Text notes added about the call	“This transcript worked well, there were only a few errors”.
mobizen_notes	Text notes added about the Mobizen screen recording	“The recording for this call started 5 seconds early”.

2.3.3 Administrator Tab

The Administrator tab allows the user to perform administrative tasks such as verifying successful connectivity to and configuration of the STT engines and creating user accounts. The following subsections describe each of these functions.

2.3.3.1 Group Management

Each user in the system is assigned to a group. The device user can only see calls made by a member of their group in the Research Portal. This eliminates any concerns about shared data when more than one group (or research team) uses an ACE Quill server. For example, if two research groups are created (Team 1 and Team 2), members of each team will only see calls initiated by their team members. This feature allows multiple teams to work independently without inadvertently sharing data. Clicking on the Group Management link opens the page for managing groups. Figure 14 shows the Manage Groups page.

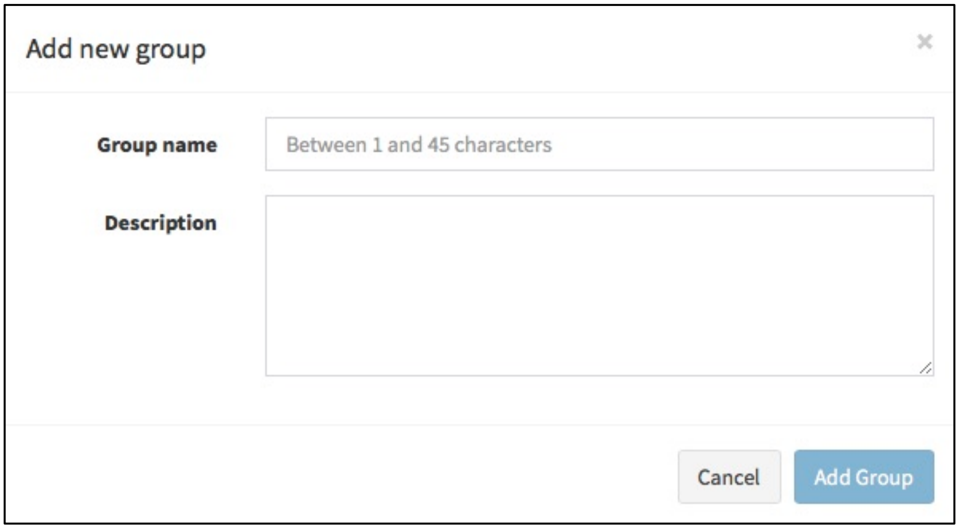


Group Name	Description
MITRE	Group for MITRE testers

Showing 1 to 1 of 1 entries

Figure 14. Screenshot of Manage Groups Page

To add a new group to the system, press the “Add New Group” button highlighted in the dotted-line box in Figure 14. This action opens the modal for Group Name and Description shown in Figure 15. A Group Name must be between one and forty-five characters.



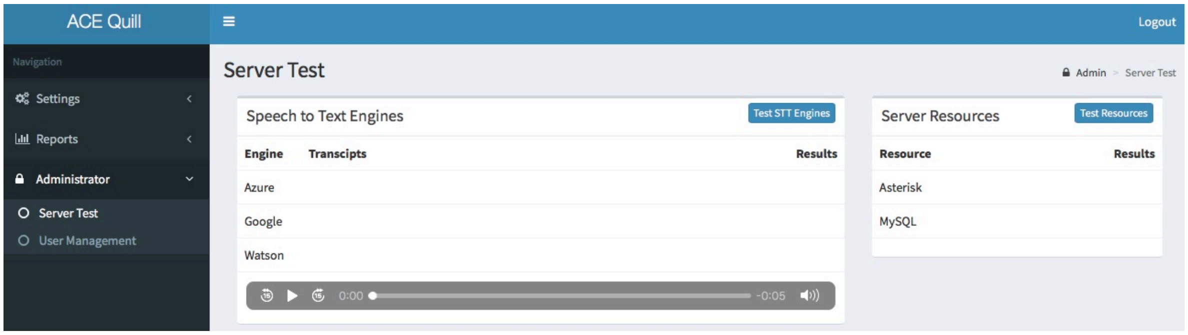
Group name: Between 1 and 45 characters

Description:

Figure 15. Screenshot of New Group Name

2.3.3.2 Server Test

The Server Test page contains two administrative tools—one for testing the STT engines and the other for testing the local servers. Figure 16 shows an example of the Server Test page.



Engine	Transcipts	Results
Azure		
Google		
Watson		

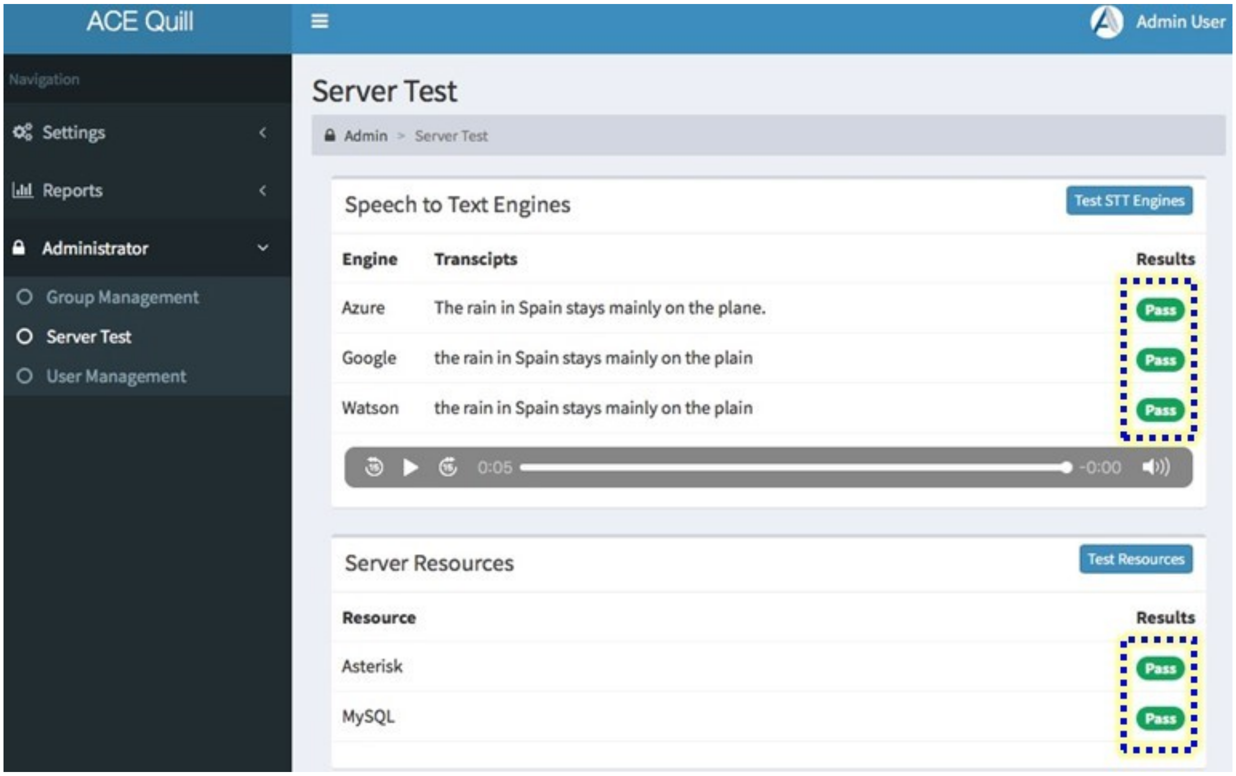
Resource	Results
Asterisk	
MySQL	

Figure 16. Screenshot of Server Test Page

The STT engine test runs an automated test to each of the three STT engines to verify configuration and connectivity. Clicking on the "Test STT Engines" button sends a simple WAV file (containing the refrain of "The rain in Spain stays mainly on the plain") to each of the providers and displays the results.

The Server Resources test verifies that the local Asterisk and MySQL servers are running and accessible. Click on the "Test Resources" button to start the server test.

Figure 17 shows the results of successful STT engine and local server resource tests (indicated by the presence of the transcript text and the green "Pass" notifications highlighted by the dotted-line boxes in the figure).



Engine	Transcipts	Results
Azure	The rain in Spain stays mainly on the plane.	Pass
Google	the rain in Spain stays mainly on the plain	Pass
Watson	the rain in Spain stays mainly on the plain	Pass

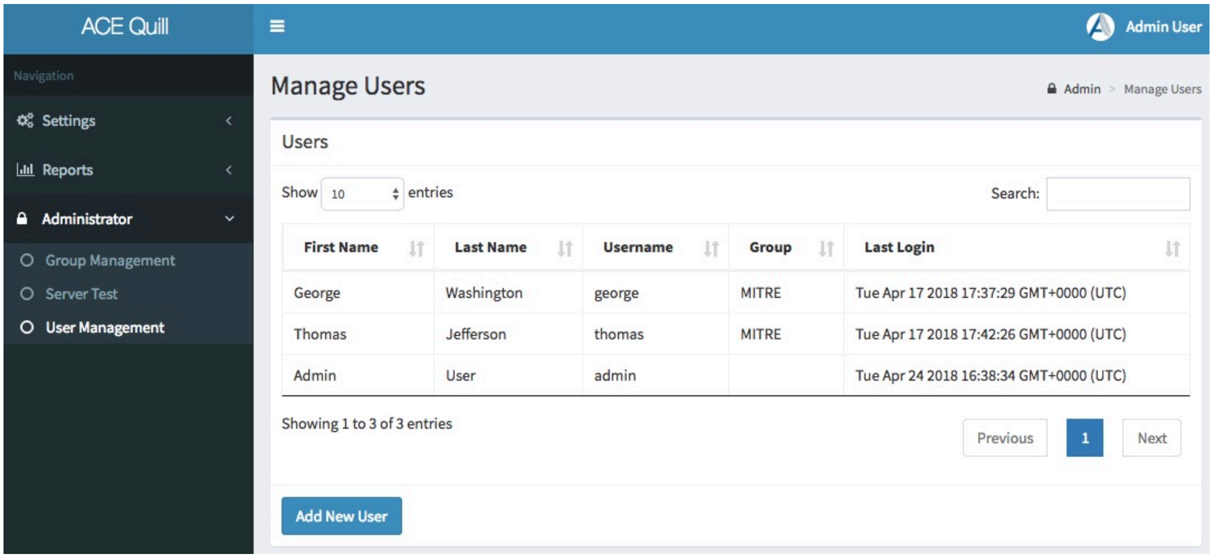
Resource	Results
Asterisk	Pass
MySQL	Pass

Figure 17. Screenshot of Successful Server Test Results

2.3.3.3 User Management

The User Management page allows the administrative user to add or remove users at the Research Portal. The portal's list of users is empty when the system is first installed. Figure 18 shows a screenshot of the initial list of users. Each user account takes up one row in the table.

Note that each username is also associated with a group. Calls made by a research user are visible to other users in the same group. This feature of the Research Portal avoids any potential issues of sharing data between research teams.

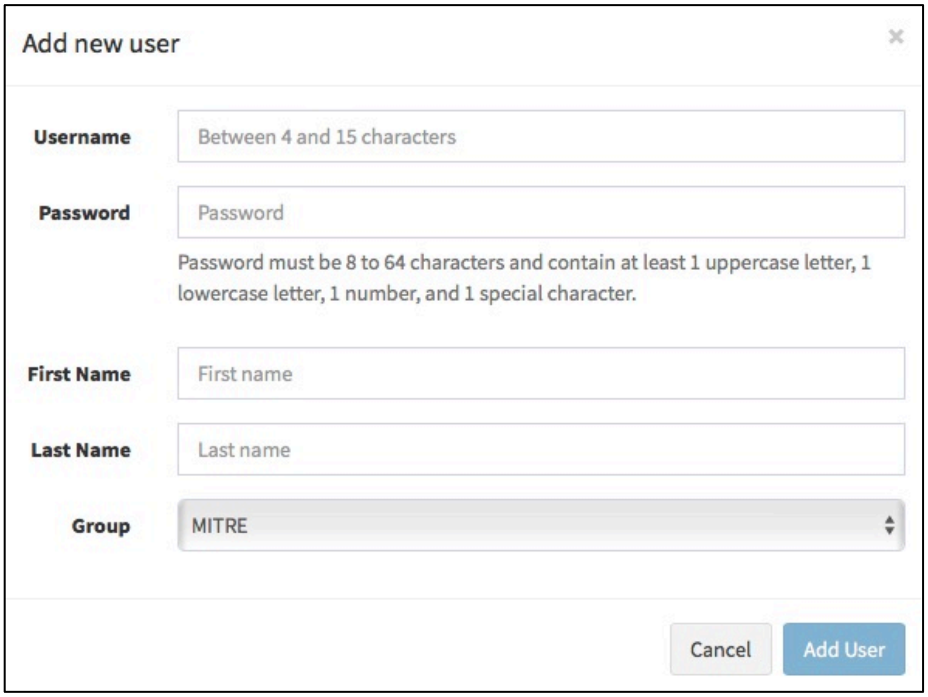


First Name	Last Name	Username	Group	Last Login
George	Washington	george	MITRE	Tue Apr 17 2018 17:37:29 GMT+0000 (UTC)
Thomas	Jefferson	thomas	MITRE	Tue Apr 17 2018 17:42:26 GMT+0000 (UTC)
Admin	User	admin		Tue Apr 24 2018 16:38:34 GMT+0000 (UTC)

Showing 1 to 3 of 3 entries

Figure 18. Screenshot of User Management Page

Selecting the "Add New User" button opens the modal shown in Figure 19. Required user information includes username, password, first and last name, and group. Note that the group must be created first before assigning a new user to a group. Once the required information has been entered, press the "Add User" button.



Add new user

Username: Between 4 and 15 characters

Password: Password

Password must be 8 to 64 characters and contain at least 1 uppercase letter, 1 lowercase letter, 1 number, and 1 special character.

First Name: First name

Last Name: Last name

Group: MITRE

Figure 19. Screenshot of Add New User Dialog

Adding a new user will insert a new row in the users table. To delete a user from the system, click on a row and select the option to delete.

3. Installation and Configuration

3.1 ACE Quill Android App

The ACE Quill client is a mobile Android application built using the Apache Cordova[1] open-source mobile development framework. Apache Cordova allows a developer to use the same source code base and build applications for different mobile platforms (e.g., Android and iOS). The client is written in TypeScript and uses Embedded JavaScript (EJS) templates. For this development effort, only an Android app was produced.

Because Apache Cordova supports multiple development platforms, the IP CTS client Android app may be built on macOS, Linux, or Windows. The MITRE development team only built and tested the Android version. Minor changes may be necessary for other platforms.

3.1.1 Installation of Required Tools

This subsection provides an overview of the ACE Quill client along with installation and setup documentation.

To build the ACE Quill client application, several tools must be installed on a development machine (macOS, Windows, or Linux). Install the following tools in the order listed:

- **Java Standard Edition Development Kit** – The standard software development environment for developing Java applications, available from Oracle.
  - Install version 8
  - Additional information is available here: http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
- **Android Studio** – The official integrated development environment (IDE) for Android, available from Google.
  - Additional information is available here: https://developer.android.com/studio/index.html
- **Node.js** – JavaScript runtime used by the build tools and Cordova.
  - Install version >= 8
  - Additional information is available here: https://nodejs.org/en/
- **Yarn** – Manages installation of JavaScript packages (alternative to npm).
  - Install from the Yarn website (not using npm)
  - Additional information is available here: https://yarnpkg.com/en/
  - MacOS users should use Homebrew (https://brew.sh) for installation
- **Cordova** – The Apache Cordova mobile application framework.
  - Install by running 'npm install -g cordova'

[1] https://cordova.apache.org

  - Additional information is available here: https://cordova.apache.org
- **Gulp** – Toolkit used to compile and package code.
  - Install by running ‘npm install -g gulp’
  - Additional information is available here: https://gulpjs.com

3.1.2 Download the Client Source Code

1. Open command line/terminal.
2. Change where you want to store the project. In the command line/terminal type:

   ```
   git clone https://github.com/FCC/ace-quill-app.git
   ```

3. Then change directory to the project directory via:

   ```
   cd ace-quill-app/android
   ```

3.1.3 Building the ACE Quill Client

Due to the large size of the Cordova build, the client is built in two steps. The first step is to establish the Cordova “skeleton” or template project. The second step repeatedly builds and copies into the template to make mobile client builds.

Before attempting to build the ACE Quill client, verify the following:

- Network proxy settings (if applicable) are correct because gradle, maven, yarn, and Cordova will need to pull code and data from the Internet
- ANDROID_HOME environment variable is set and points to the correct directory

The first time you prepare this environment, you will need to install the Cordova template. In the top-level directory (macOS or Linux), type:

```
cordova-install.sh
```

On Microsoft Windows, type:

```
./cordova-install.bat
```

In the top-level directory, type the following to install all necessary packages:

```
yarn install
```

To build the client, run the following commands:

```
gulp build
cd cordova/ACEQUILL
rm -rf platforms
cordova platform add android
cd ../..
gulp build
cd cordova/ACEQUILL
```

```
    cordova build
```

This will generate an APK file whose location will be displayed as the output of the final “cordova build” output.

3.1.4 Installing the Client

Because the application is not currently in the Google Play store, the phone must be configured in developer mode before installing the app onto the Android phone. **Note:** The phone must be secured with a password to assure secure storage of credentials and proper operations.

The following instructions were developed using a Google Pixel running Android version 8.x. The instructions may vary for other versions of Android.

Uninstall the current version of the ACE Quill app from the Android phone (if installed):

1. Long-press the app icon
2. Drag to “remove” in the upper left-hand corner

Enable USB debugging on Android phone:

1. On the Android phone, go to Settings > Developer Options > USB debugging
2. Toggle on

Enable file transfer on Android phone:

1. On the Android phone, swipe down from the top of the homepage
2. Tap the Android System notification (which may say "USB for charging")
3. Select use USB for "Transfer files (MTP)"

Allow Android apps from outside of the Google Play Store:

1. On the Android phone, go to Settings > Security (may be called "Lock Screen and Security")
2. Toggle on "Unknown sources"

(For macOS) Upload the Android app in Android PacKage (APK) format to the Android phone from your computer:

1. On the Mac, download and/or open "Android File Transfer" app
2. Connect phone to computer via USB or USB-C cable
3. In Android File Transfer, navigate to the "Download" folder
4. Drag the APK file from your computer into the "Download" folder

(For Windows) Upload debug APK to Android phone from your computer:

1. Connect the phone to the computer using a USB or USB-C cable
2. Select "File Transfer" on the notification window

3. On the computer in Windows Explorer, navigate to Pixel XL > Internal Storage > Download
4. Drag the APK file into the Download folder

Install debug APK on Android phone:

1. On the Android phone, go to Settings > Storage > Explore
2. Click on the debug APK file
3. Install

Modify permission after install on the Android phone:

1. Open settings > apps > ACE Quill
2. Select "Permissions"
3. Toggle on the required permission options (Camera & Microphone)

Add App Icon to Homepage on Android phone:

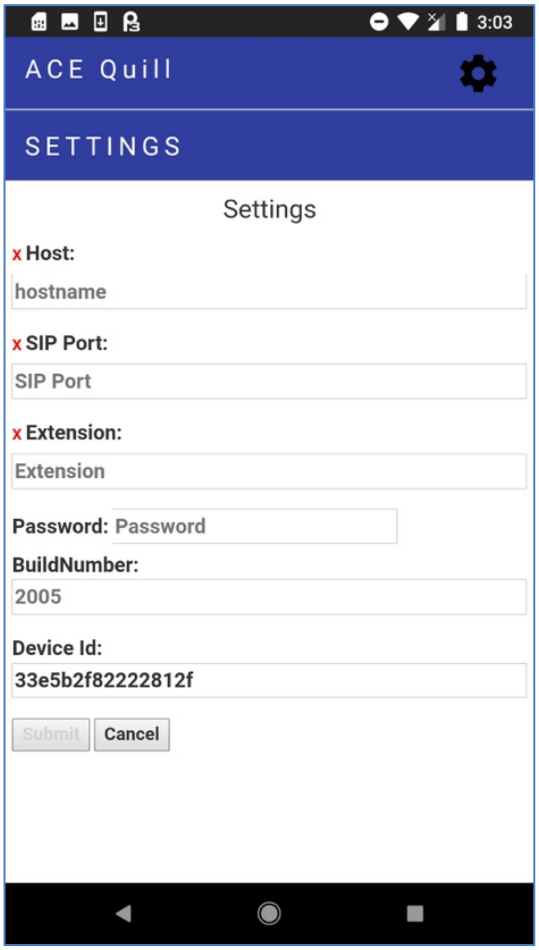
1. Go to homepage
2. Swipe up from the bottom of the page to display the apps drawer
3. Long-press the app icon you want to add to the Home screen
4. Drag the app to the Home screen page

3.1.5 Configuring the Client

Once the client is installed onto an Android device, you must configure settings on the client before the client can place a call.

On the client, select the gear (upper right), then select Settings and you will see the screen shown in Figure 20. Fill out the form with parameters provided by the system administrator and select the Submit button to go to the opening page of the app. The required parameters are:

- **Hostname –** The full hostname (including domain) of the server running the ACE Quill server software
- **SIP Port –** The port for SIP communications
- **Extension –** The unique phone extension (e.g., 6000, 6001, …) assigned to this Android phone
- **Password –** The Asterisk password



Settings

x Host: hostname

x SIP Port: SIP Port

x Extension: Extension

Password: Password

BuildNumber: 2005

Device Id: 33e5b2f82222812f

Figure 20. Screenshot of IP CTS Client Settings

3.2 ACE Quill Server

The ACE Quill Server is a component of the ACE Quill Prototype. The ACE Quill server was developed in Amazon Web Services (AWS) on a CentOS 7.4 Linux instance running on a t2.medium Elastic Compute Cloud (EC2) instance (a local server was an alternative). The CentOS Linux installation instructions can be found on the CentOS website.

To download the ACE Quill server software, follow these steps:

1. Open command line/terminal.
2. Change where you want to store the project. In the command line/terminal type:

```
git clone https://github.com/FCC/ace-quill/ace-quill-server.git
```

3. Then change directory to the project directory via:

```
cd ace-quill-server/node-server
```

The ACE Quill Server performs two main tasks:

- Provides a web-based interface for researchers and administrators to configure STT engines, manage user accounts, and review research data
- Records a call on the Asterisk PBX as a WAV file, sends the WAV file to a STT engine, and sends transcribed text to the IP CTS Client application using SIP messages

3.2.1 Asterisk Private Branch Exchange Installation

The Asterisk PBX is included as part of ACE Quill server downloaded in section 3.2 in the ace_quill_server/asterisk folder. Installation instructions can be found in the README.md file.

The Asterisk PBX is installed with a script that updates the configuration files with the required parameters. The installation script is located in ~/acequill_server/asterisk/scripts/ACEQUILL_asterisk_install.sh.

A Session Traversal of User Datagram Protocol (STUN) server is required by Asterisk. The STUN server allows network address translation (NAT) clients to set up phone calls to a Voice Over Internet Protocol (VoIP) provider outside of the local network. A publicly available STUN server (e.g., hosted by Google) can be used.

The following parameters are required by the Asterisk installation script:

- Public IP address
- Private/local IP address
- Private/local IP netmask
- Fully qualified domain name used by the SIP trunk provider (e.g., fqdn.twilio.com)
- Certificate
- Key
- STUN server hostname
- STUN port

To run the script:

```
cd ace_quill_server/asterisk

cd scripts

chmod 777 ace_quill_asterisk_install.sh

./ace_quill_asterisk_install.sh –public-ip <public IP address> --local-
ip <local IP address> --local-ip-mask <local IP netmask> --twilio-fqdn
<fqdn> --crt-file <path to certificate> --crt-key <path to key> --stun-
server <fqdn> --stun-port
```

The ACE Quill Asterisk server configuration uses Elastic SIP Trunking. Elastic SIP Trunking is a cloud-based service that connects IP-based communications with the PSTN, which allows the ACE Quill prototype to call standard phones. Prototype administrators can select from a number of Elastic SIP Trunk vendors, and other vendors would provide similar configuration options.

The following instructions are based on a Twilio account:

- Sign into your Twilio account

- Select the button with three dots under the home button
- Select "Elastic SIP Trunking”
- Select “Trunks”, “Create new SIP Trunk”
  - Enter the subdomain for your Asterisk server in the “Friendly Name” field then click "Create"
- Select "Termination" in the menu on the left
  - In the "Termination SIP URI" input box, enter the subdomain for your server
- In the "Authentication" section, click on the plus-sign button to the right of "IP Access Control Lists"
  - Use the subdomain to name the ACL, then enter the elastic IP address of your server in the list
- Click on "Origination" to the left
  - Select “Add new Origination URI”
    - This should either be "sip:address", where address is either the FQDN or the elastic IP address of your server
- Click on "Numbers" to the left
  - Select “Buy a Number” to purchase and associate a new ten-digit number to the SIP trunk

3.2.1.1 Asterisk Server Operation

The Asterisk server is a system service that can be started and stopped with the Linux service command. To launch the Asterisk server, run the following command:

```
sudo service asterisk start
```

To stop the Asterisk server, run the following command:

```
sudo service asterisk stop
```

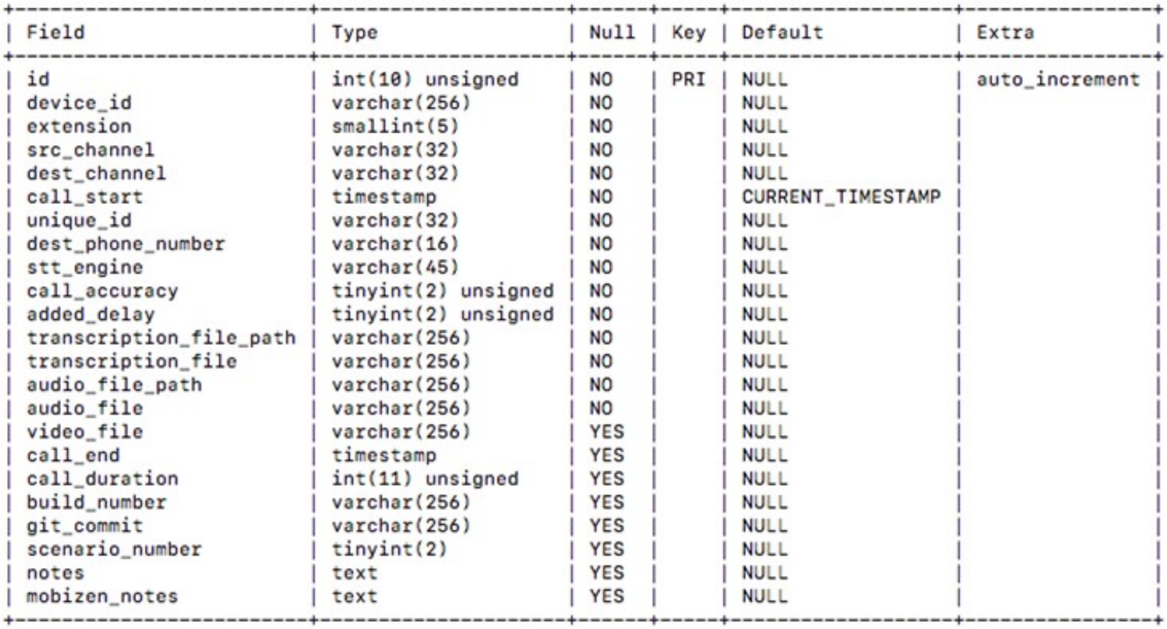
3.2.2 MySQL Database

The ACE Quill prototype uses the MySQL database to store several types of data. For example, MySQL houses all username and passwords, research, and call data.

The MySQL database schema is included as part of ACE Quill server downloaded in section 3.2. The filename is ace_quill.sql in the ace_quill_server/node_server folder.

Installation instructions can be found in the README.md file.

The research data displayed in the Call Logs table is a subset of the MySQL data stored per call. The actual database contains the fields shown in Figure 21. All research data are available for download as a CSV file. The CSV file will contain every column shown in Figure 21.



```
+-------------------------+---------------------+------+-----+-------------------+----------------+
| Field                   | Type                | Null | Key | Default           | Extra          |
+-------------------------+---------------------+------+-----+-------------------+----------------+
| id                      | int(10) unsigned    | NO   | PRI | NULL              | auto_increment |
| device_id               | varchar(256)        | NO   |     | NULL              |                |
| extension               | smallint(5)         | NO   |     | NULL              |                |
| src_channel             | varchar(32)         | NO   |     | NULL              |                |
| dest_channel            | varchar(32)         | NO   |     | NULL              |                |
| call_start              | timestamp           | NO   |     | CURRENT_TIMESTAMP |                |
| unique_id               | varchar(32)         | NO   |     | NULL              |                |
| dest_phone_number       | varchar(16)         | NO   |     | NULL              |                |
| stt_engine              | varchar(45)         | NO   |     | NULL              |                |
| call_accuracy           | tinyint(2) unsigned | NO   |     | NULL              |                |
| added_delay             | tinyint(2) unsigned | NO   |     | NULL              |                |
| transcription_file_path | varchar(256)        | NO   |     | NULL              |                |
| transcription_file      | varchar(256)        | NO   |     | NULL              |                |
| audio_file_path         | varchar(256)        | NO   |     | NULL              |                |
| audio_file              | varchar(256)        | NO   |     | NULL              |                |
| video_file              | varchar(256)        | YES  |     | NULL              |                |
| call_end                | timestamp           | YES  |     | NULL              |                |
| call_duration           | int(11) unsigned    | YES  |     | NULL              |                |
| build_number            | varchar(256)        | YES  |     | NULL              |                |
| git_commit              | varchar(256)        | YES  |     | NULL              |                |
| scenario_number         | tinyint(2)          | YES  |     | NULL              |                |
| notes                   | text                | YES  |     | NULL              |                |
| mobizen_notes           | text                | YES  |     | NULL              |                |
+-------------------------+---------------------+------+-----+-------------------+----------------+
```

Figure 21. MySQL Reference Data Table

This subsection provides an overview of the MySQL server along with installation and setup documentation.

Install the dependent software packages using yum:

```
sudo yum install unixODBC unixODBC-devel libtool-ltdl libtool-ltdl-
devel mysql-connector-odbc
```

Download the Yum repository configuration file:

```
wget https://dev.mysql.com/get/mysql57-community-release-el7-
9.noarch.rpm
```

Install the Yum repository configuration file:

```
sudo rpm -ivh mysql57-community-release-el7-9.noarch.rpm
```

Install the MySQL server:

```
sudo yum install mysql-server
```

Start the MySQL server:

```
sudo systemctl start mysqld
```

A temporary password is generated as part of the installation process. It is in the mysqld.log file and can be found with this command:

```
sudo grep 'temporary password' /var/log/mysqld.log
```

MySQL includes a security script to change some of the default, non-secure options:

```
sudo mysql_secure_installation
```

As part of the security script, the user will be prompted for a new password. The new password must be at least 12-characters and contain at least one uppercase letter, one lowercase letter, one number and once special character.

To stop the MySQL server:

```
sudo systemctl stop mysqld
```

3.2.2.1 MySQL Database Configuration

The IP CTS prototype will require two MySQL databases and creation of the corresponding accounts according to the following instructions:

Enter the MySQL shell, enter the root password when prompted:

```
mysql -uroot -p
```

Inside the MySQL shell, create the following databases and accounts for access:

```
mysql> create database asterisk;


mysql> create database ace_quill;
```

Create the database accounts and issue the grants:

```
mysql> grant all on asterisk.* to 'ace_quill'@'localhost' identified by
'password here';


mysql> grant all on ace_quill.* to 'ace_quill'@'localhost' identified
by 'password here';
```

Exit the MySQL shell:

```
mysql> exit;
```

The next step is to instantiate the databases. The asterisk_schema.txt and ace_quill_schema.txt files are included as part of the public release download.

From the command prompt (not the MySQL prompt) type the following and enter the password when prompted:

```
mysql -uace_quill -p asterisk < asterisk.sql


mysql -uace_quill -p ace_quill < ace_quill.sql
```

3.2.2.2 Running the MySQL Server

To start the MySQL server:

```
sudo systemctl start mysqld
```

To stop the MySQL server, run the following command:

```
sudo systemctl stop mysqld
```

3.2.3 Node.js Server Installation

The ACE Quill server has several software dependencies that must be installed. To install the ACE Quill software and the required dependencies, follow these steps:

- `git clone https://github.com/FCC/ace-quill-server.git`
- Install Node.js >= version 8.x
  - Visit https://nodejs.org/en/
- Install Bower >= v1.8.0
  - `npm install -g bower`
  - `cd acequill-server`
- Install the required Node.js modules
  - `npm install`
- Install the required Bower packages
  - `bower install`
- Install PM2 >= v2.4.6
  - `npm install -g pm2`

3.2.3.1 Secure Sockets Layer Configuration

The ACE Quill software uses Secure Sockets Layer (SSL) to ensure secure communications between the server and the browser. SSL requires a valid key and certificate as part of the configuration.

To enable SSL in the IP CTS server, specify ‘https’ as the protocol value in the config.json file. The location of the SSL key and certificate can be specified in the config.json by using the ssl:cert and ssl:key parameters in the form of folder/file (e.g., ssl/mycert.pem and ssl/mykey.pem).

3.2.4 Speech-to-Text Engine Configuration

The ACE Quill prototype was developed to support interfaces to the following STT engines:

- Google Speech (https://cloud.google.com)
- IBM Watson (https://www.ibm.com/watson/services/speech-to-text/)
- Microsoft Bing Speech (https://azure.microsoft.com/en-us/services/cognitive-services/speech/)

Follow the link for each STT engine, create an account, and download the account credentials as a JSON file.

3.2.4.1 Microsoft Bing Speech

The Microsoft Bing Speech library can use a network proxy (if one is required for external network access). The configuration file (config.json) has the following proxy definition:

```
"proxy":"http://<IP address>:<port>/"
```

Verify with the local network administrator if a proxy is required.

3.2.4.2 Google Speech

Two environment variables are required to contact the Google Speech server:

```
export GOOGLE_APPLICATION_CREDENTIALS = <path to google.json>

export GCLOUD_PROJECT='<account credentials here>'
```

3.2.4.3 IBM Watson

If behind a proxy, Watson uses a web socket connection to 'wss://stream.watsonplatform.net/speech-to text/api' for its STT services. If the ACE Quill server is running behind a proxy, the URL will need to be whitelisted or a modification to the Watson-developer-cloud node module will be required, as follows:

```
cd node_modules/watson-developer-cloud/speech-to-text

vi recognize_stream.js
```

Search for the ‘W3CWebSocket’, which block should be similar to the following:

```
// node params: requestUrl, protocols, origin, headrs,
      extraRequestOptions

// browser params: requestUrl, protocols (all others ignored)

const socket = (this.socket = new W3CWebSocket(

    url,

    null,

    null,

    options.headers,

    null

));
```

Create a ‘proxy-agent’, then add it as the extraRequestOptions:

```
//Server requires proxy

const ProxyAgent = requir('proxy-agent');

const extraRequestOptions = {

 agent: new ProxyAgent("<your proxy address>")

};

// node params: requestUrl, protocols, origin, headers,
      extraRequestOptions
```

```
// browser params: requestUrl, protocols (all others ignored)
const socket = (this.socket = new W3CWebSocket(
    url,
    null,
    null,
    options.headers,
    extraRequestOptions  //was null
));
```

3.2.4.4 Server Software Configuration

The ace_quill_server directory contains a file named config.json. The config.json file contains the configuration parameters used by the server. The configuration parameters and the definitions are as follows.

```
{
    "debuglevel": "ALL | TRACE | DEBUG | INFO | WARN ERROR |      FATAL
| OFF",
    "port": server listen port #,
    "ssl" : {
      "key": "path to key",
      "cert": "path to cert"
    },
    "session": {
      "name":"session name",
      "resave": "true",
      "saveUninitialized": "true",
      "secret": "secret here",
      "secure": "true"
    },
    "sttService": "enabled | disabled",
    "transcriptFilePath": "path to transcripts directory",
    "wavFilePath": "path to wav file directory",
    "videoFilePath":"path to uploads directory",
    "mysql": {
      "host": "localhost",
      "user": "mysql username",
      "password": "mysql password",
      "database": "ace_quill"
    },
    "asterisk": {
      "port": 5038,
      "host": "IP address of localhost",
      "user": "asterisk username",
      "password": "asterisk password"
    }
}
```

3.2.4.5 Starting the Server and Accessing the Research Portal

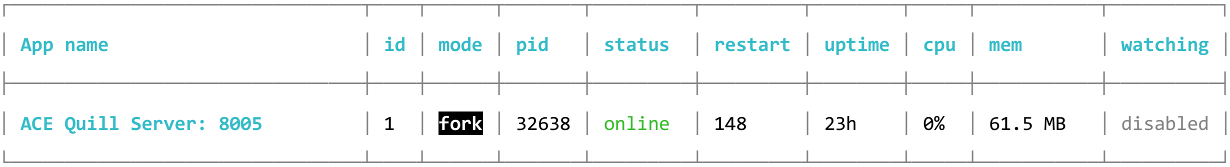
Once the config.json file has been properly configured, the ACE Quill server can be started by running the following command:

```
cd ace_quill_server
pm2 start process.json
```

To check the status of the running server, run the following command:

```
pm2 status
```

The output should be similar to the following (the status should be "online") in Figure 22:



```
┌──────────────────────────┬────┬──────┬───────┬────────┬─────────┬────────┬─────┬─────────┬──────────┐
│ App name                 │ id │ mode │ pid   │ status │ restart │ uptime │ cpu │ mem     │ watching │
├──────────────────────────┼────┼──────┼───────┼────────┼─────────┼────────┼─────┼─────────┼──────────┤
│ ACE Quill Server: 8005   │ 1  │ fork │ 32638 │ online │ 148     │ 23h    │ 0%  │ 61.5 MB │ disabled │
└──────────────────────────┴────┴──────┴───────┴────────┴─────────┴────────┴─────┴─────────┴──────────┘
```

Figure 22. PM2 Status Output

To access the research portal, the resulting URL will be in the following format:

- https:/<your /host name>:<your config port>/admin

The ACE Quill server provides the connections to the STT engines. The server must be operational before running the ACE Quill client app.

4. Troubleshooting

This section contains tips to troubleshoot issues with the system.

Issue: Cannot place a call from the app

- Check the permissions for the app.
  - Long press the app icon, select view app info, select permissions, enable camera, microphone, and access to contacts.
- Check the settings to ensure host, extension, and password are correct.
  - Extension and password should be configured in the Asterisk sip.conf file.
  - Contact the Asterisk system administrator if you are unsure of these values.
- Ensure phone has network access
  - Phone requires Wi-Fi or Long-Term Evolution (LTE) network access

Issue: Call connects but transcripts are not generated

- Verify that the ACE Quill server app is running.
  - Open a Secure Shell (SSH) to the server and run “pm2 status” to check the server status.
- Verify that the STT engines are available.
  - Use the ACE Quill administrative portal to run the self-tests.
- Check the STT Configs on the admin portal to ensure the credentials are correct.

Issue: IP CTS Server node app will not start

- Check if the port specified in the config.json file is in use.
  - On Linux, run “netstat -tan |grep <port number>”
- Perform a 'npm install' to ensure all node modules are installed, see README.md.

Issue: IP CTS Admin page is accessible, but is missing style sheets and other dependencies.

- Run 'bower install' to install any client-side libraries, see README.md.

Issue: App is not saving settings

- The app requires the device to configure a lock screen for security.

Issue: Help, I forgot the password for the admin account

- Unfortunately, there is no password recovery.
- To access the administrator-privileged page, you will need to delete the admin account from MySQL and then navigate to https://host:port/firsttimesetup to create a new admin account.

Acronyms

API	Application Programming Interface
APK	Android PacKage
ASR	Automatic Speech Recognition
AWS	Amazon Web Services
CAMH	CMS Alliance to Modernize Healthcare
CDR	Call Detail Record
CMS	Centers for Medicare & Medicaid Services
COE	Center of Expertise
CMS	Centers for Medicare & Medicaid Services
CSV	Comma Separated Value
CTS	Captioned Telephone Service
EC2	Elastic Cloud Compute
EJS	Embedded JavaScript
FAR	Federal Acquisition Regulation
FCC	Federal Communications Commission
FFRDC	Federally Funded Research and Development Center
HHS	Department of Health and Human Services
HoH	Hard of Hearing
IDE	Integrated Development Environment
IP	Internet Protocol
JSON	JavaScript Object Notation
LTE	Long Term Evolution
NAT	Network Address Translation
NTL	National Test Lab
PBX	Private Branch Exchange
PSTN	Public Switched Telephone Network
SIP	Session Initiation Protocol
SSH	Secure Shell
SSL	Secure Sockets Layer

STT	Speech-to-Text
STUN	Session Traversal Utilities for NAT
TRS	Telecommunications Relay Service
URL	Universal Resource Locator
VoIP	Voice over IP
VRS	Video Relay Service

Notice

This (software/technical data) was produced for the U. S. Government under Contract Number HHSM-500-2012-00008I, and is subject to Federal Acquisition Regulation Clause 52.227-14, Rights in Data-General.

No other use other than that granted to the U. S. Government, or to those acting on behalf of the U. S. Government under that Clause is authorized without the express written permission of The MITRE Corporation.

For further information, please contact The MITRE Corporation, Contracts Management Office, 7515 Colshire Drive, McLean, VA 22102-7539, (703) 983-6000.