TABLE OF CONTENTS

ABSTRACT

The evolution of voice control within the last few years has introduced the connected home to a new form of convenience and efficiency. With this increased attention on voice control from homeowners, designers, architects, and home technology professionals, it is critical that programming standards are set for a voice-friendly experience. This white paper provides:

- An overview of Josh.ai's proprietary Natural Language Processing (NLP)
- How Lutron systems integrate with Josh.ai
- Programming best practices to provide a successful foundation for voice control of environmental devices
- Control capabilities of Lutron systems and devices via Josh.ai
- Benefits of a voice-controlled environment

The objective of this paper is to put forth a standard for voice-controlled environments that will aid in successful Lutron project deployments and positive user experiences.

EXECUTIVE SUMMARY

Voice control's proliferation in the smart home is making it more important than ever to know how to optimize Lutron systems for the highest degree of interoperability.

Today, the mass market voice assistant solutions rely on a skill or scene-based approach, with trigger phrases that need to be repeated exactly as dictated. Since these assistants try to handle a wide variety of tasks outside the scope of home control, it is difficult for them to deliver accuracy around mishearings or mis-translations of commands. The resulting experience is very rigid, with each command acting as a verbal button press, and the assistant not knowing the device types being controlled or the change in states of those devices.

Conversely, Josh.ai is purpose-built with a contextual understanding of smart home control. Deep Lutron integration enables Josh.ai to limitlessly scale and differentiate what types of devices are controllable, where they are located, what they are named, and their current state. This contextual awareness, coupled with features like memory in the software for the most recent command and room-awareness, provides a more natural experience that reduces the user's need to memorize activation phrases verbatim.

Josh.ai communicates with the Lutron processor over the local home network. Once Josh.ai and Lutron have authenticated with each other, Josh.ai will learn the configuration of the home from Lutron. Josh.ai starts pulling in floors, rooms, and device types. Since Josh.ai is communicating directly to each device in a project, good Lutron programming is necessary to ensure a user-friendly experience. After an install is complete, it is then a critical support procedure to continue editing room and device aliases based on a user's chat history. By continuously monitoring how clients refer to their environment or how an accent is possibly mistranslated from speech-to-text by the Automatic Speech Recognition (ASR) platform, we can ensure that the system continues providing as accurate of an experience as possible.

JOSH.AI - NATURAL LANGUAGE PROCESSING

What is Natural Language Processing (NLP):
NLP, is a branch of artificial intelligence that encompasses the interaction between computers and humans using natural speech. The objective of NLP is to interpret the ambiguous nature of human language by applying algorithms to extract meaning through syntax and semantics.

The analysis of syntax and semantics are the two primary techniques that lead to the understanding of natural language. Syntax analysis is the process of parsing language within the rules of formal grammar, while semantic analysis is the process of understanding the meaning being conveyed through verbiage and sentence structure.

In Josh.ai's case, its NLP is geared towards how a user would want to communicate to their home. Similar to a human sitting in the same room, Josh.ai understands what is accessible to control and the feature-sets associated with each device-type. To add to its contextual awareness, Josh.ai also knows where devices are relative to the user's location, the current state of the home, and the device or subset of devices that were previously controlled.

Anaphora:
Josh.ai remembers what device you last activated in order to reduce redundant language and commands. By utilizing anaphora, Josh.ai enables users to give a command like, *"Ok Josh, dim the lights in the Living Room"* and follow it up with *"Ok Josh, raise them 25%"* to adjust the Living Room's lights.

Anaphora is also built into the touch capacitive LED dial of Josh Micro, providing intuitive manual control for devices that were previously spoken to. The dial functions as a contextual slider for approximately 30 seconds after a command is given to lighting or shades before reverting to its default setting (see next page):

- Following a command to lights, like *"Ok Josh, raise the lights,"* the Micro LED dial will animate yellow to signal that it's able to adjust the brightness of the lights that were just activated.

- For color oriented commands to Ketra lights, like *"Ok Josh, make the lights turquoise,"* the Micro dial will illuminate a rainbow animation to allow manual alteration of light color.

- After giving a command to shades, like *"Ok Josh, lower the blinds,"* the Micro dial will display a blue animation to signal that the dial is now able to raise or lower the subset of blinds that were just activated.

Room-Awareness with Josh Micro:
A development that has increased ease of use with Josh Micro is its recognition of the room and devices it has immediate control over. This simple, yet robust feature eliminates the need for users to remember the name of the room they are currently located in. It also significantly reduces the verbiage in commands so that users do not have to reiterate, *"Ok Josh, turn on the lights in the*

kitchen and open the shades in the kitchen." Instead, someone can simply tell the Micro assigned to their kitchen to, *"Ok Josh, turn on the lights and open the shades."*

Of course, a user can give a command to other rooms or devices from any Micro depending on its permission settings. However, they will need to be more specific with regards to room or device names when telling Josh.ai to activate something outside of the room they are currently in.

JOSH.AI + LUTRON INTEGRATION

Homeworks QS, RadioRA2, and Connect Bridge:

- Hub Discovery
    - Josh.ai uses Lutron proprietary discovery.
    - Extracts project name from the discovery response.
    - In order to support large residences that have multiple Lutron processors per project, Josh.ai chooses a single processor to communicate with.
    - If a processor falls offline, Josh.ai associates devices with a different processor within the project.
    - Authorization is shared across processors within a project.
    - Josh.ai attempts to automatically authorize projects that support the default telnet credentials.
- Device Discovery
    - Performed when Josh.ai is connected to the network to make sure it discovers the latest devices and whenever the project is modified.
    - Josh.ai parses devices from the configuration XML.
    - When areas in the configuration XML contain supported devices, Josh.ai auto-creates rooms and places devices into each room.
    - Josh.ai discovers:
        - Lights
        - Switches
        - Garage doors
        - Gates
        - Shades
        - Fans
        - Ketra (Homeworks QS with Connect Bridge only)
        - Thermostats
        - Keypads
        - Phantom Keypads (Homeworks QS only)
- Lights:
    - Supported capabilities are automatically determined:
        - On/off
        - Brightness

        - Color (Ketra*)
        - Color temperature (Ketra*)
        - Vibrancy (Ketra*)
- Shades:
    - Supported capabilities are automatically determined:
        - Up/down
        - Adjustable by percentage
- Garage Doors/Gates:
    - Pulsed CCOs with *"Garage"* or *"Gate"* in their name are discovered as device-type Garage Doors within Josh.ai
- Fans:
    - Capabilities are automatically determined.
    - Exhaust fans are discovered as binary.
    - Ceiling fans support five speeds.
- Fireplaces:
    - Switches programmed with "Fireplace" in their name are discovered as a device-type Fireplace within Josh.ai
    - Capabilities are on/off
- Keypads:
    - Keypad button presses can be tapped manually via the Josh.ai device tile.
    - Keypad button presses can be programmed into Josh.ai scenes for verbal recall.
    - Button names update when they are changed in the Lutron project.

QSX, RA2 Select, and Caseta:

- Hub Discovery
    - Josh.ai uses mDNS discovery.
    - Authorization via physical button press as defined in LAP protocol.
    - The Smart Bridge Pro is necessary to communicate to Josh.ai for Caseta Systems
- Device Discovery
    - Josh.ai uses LEAP area, device, zone, button, and virtualbutton endpoints.
    - Discovery for devices and zones.
    - When areas in device/zone response contain supported devices, Josh.ai automatically creates rooms and places devices into each room.
    - Josh.ai discovers:
        - Lights
        - Shades
        - Pico remotes
        - Fans
        - Ketra (QSX only)
        - CCOs (QSX only)
        - Keypads (QSX only)
        - Phantom Keypads (QSX only)

        - Scenes (RA2 Select and Caseta only)
- Lights:
    - Supported capabilities are automatically determined:
        - On/off
        - Brightness
        - Color (Ketra*)
        - Color temperature (Ketra*)
        - Vibrancy (Ketra*)
- Shades:
    - Supported capabilities are automatically determined:
        - Up/down
        - Adjustable by percentage
- Pico Remotes and Keypads:
    - Josh.ai pulls in button engravings when possible.
- CCOs:
    - Device-type is automatically determined from the device's name for Garage Doors, Gates, and Fireplaces.

Illumination:
- Hub Discovery
    - Josh.ai uses Lutron proprietary discovery.
- Device Discovery
    - Josh.ai parses the XML pulled via FTP.
    - Discovery is done when Josh.ai connects to the network and begins communicating with the Lutron processor.
    - When areas in config XML contain supported devices, Josh.ai auto-creates rooms and places devices into their respective rooms.
    - Josh.ai discovers:
        - Lights
        - Shades
- Lights:
    - Supported capabilities are automatically determined:
        - On/off
        - Brightness
- Shades:
    - Supported capabilities are automatically determined:
        - Up/down
        - Adjustable by percentage

Best practices and troubleshooting:

For a Lutron Homeworks QS or RadioRA2 system, Josh.ai identifies the Lutron hub once it is on the network and will attempt to connect via Lutron's default login credentials. In the event that the connection is not made instantaneously, a few troubleshooting steps are:

- It is recommended to create a dedicated Telnet login credential for the Josh.ai system.
    - This credential should not be used for any other control systems. Lutron limits the number of connections each login has access to, so it is important these credentials are unique for Josh.ai.
- If the *Lutron Homeworks QS* or *Lutron RadioRA2* tab is not showing up within the Josh.ai *Authorize Devices* page, there may have been an issue discovering the hub. The most common reason Josh.ai would fail to discover the Lutron hub is when the Josh Micro is connected via WiFi. Josh.ai utilizes multicast network traffic to discover the Lutron hub, and certain network configurations can block that traffic when going from WiFi to the ethernet connected Lutron hub.
    - The simplest solution is to temporarily connect a Josh Micro to POE, either directly to a switch that provides POE or with an injector. The Lutron hub should then be discovered, at which time the Micro can be moved back to WiFi.
- The *lutron/lutron* default credentials are rejected because Josh.ai cannot get feedback when querying device state via that login. All of the devices would show as offline if it were accepted.
- Recent versions of HomeworksQS and RadioRA2 no longer accept the well known default Lutron logins. If Josh.ai cannot automatically connect to the Lutron system, it is likely that a dedicated login for Josh.ai is needed. Do not forget to transfer the project to your Lutron system!

PROGRAMMING BEST PRACTICES FOR LUTRON + JOSH.AI

Once discovered on the network, Josh.ai automatically pulls in the pre-existing Lutron programming. By considering the following naming practices, the Lutron system will be configured as robustly as possible for initial use and have a strong foundation to continue making voice-friendly improvements.

Choose names that are easy to pronounce:
For example, an alias like *"NW B-Room 2"* is not a room name that someone would think of referring to. Aliases like "*Kitchen*" and *"Living Room"* make sense and are easy to remember. The same goes for device names, *"Light 123"* is not how a user would refer to their *"Master Bathroom Sink Light."*

Choose names that will not be easily misheard:
Single syllable words can present challenges in the speech-to-text conversion process. For example, a room called "*Den*" could be misheard as "*Din*" or "*Then*," particularly if there is background noise. A room can certainly be named the "*Den*" in programming, but it is important to

think through how it could be misheard and then monitor chat history to see how it is possibly mistranslated. This is especially true if any users have accents that may need to be accounted for.

Words with multiple syllables require a higher degree of annunciation and therefore present a greater likelihood of accuracy when being sent to the ASR. Room names like, "*Master Bedroom*" or "*Dining Room*" have proven to be reliably understood despite background noise and accents.

Avoid duplicates:
Technically, the word "*Room*" does not need to be included in a room name for Josh.ai to interpret it properly. For example, if a room were called "*Living*" and someone gave a command to "*turn on the lights in the Living Room*" it would still work. That said, cases have come up when there was both a room called "*Living*" as well as a room called "*Living Room*" on a single system and that causes confusion. Avoid these sorts of duplicates.

Check spelling:
A nice thing about voice control is that a command that goes through the speech-to-text conversion process will not be misspelled. That said, if a room or device name contains a typo in the programming, it will not align when a voice command is given. For example, if a room is named "*Kicthen*" (notice the misspelling) in the Lutron programming, it will not match when a user asks Josh.ai to do something in the "*Kitchen.*"

- This naming convention holds true when dealing with abbreviations as well. For example, it is recommended to name a device "*Floor Lamp*" instead of "*Fl Lmp.*"
- Lastly, it is a good idea to add any alternatives as additional aliases within Josh.ai. For example, we as humans know "*TV Lights*" and "*Television Lights*" likely mean the same thing while Josh.ai might not always inherently make that connection.

Make rooms easy to remember:
If possible, it is best to do a walk through with the client to understand how they want to refer to rooms. Rather than picking an obscure name like "*Foyer*," if the client wants to refer to the area as the "*Entrance*" make sure to name it in a way that is clear to them.

- This often occurs with lighting loads as well - "*sconce*" and "*recessed*" might be common for an installer to refer to, but not for a client.

Check word spacing:
The best way to check if speech is being converted to text as multiple words versus a single word is to speak to a Josh Micro or the Josh.ai iOS app. The text of how the command is being translated will populate chat history and provide clarity around the ASR's interpretation of the word (or words). For example, should a scene be triggered using "*Goodnight*" or "*Good night?*" Is the proper term "*Downlight*" or "*Down light*?" For common instances like these, Josh.ai will likely handle it regardless if a space is added or not. For less common cases that Josh.ai may have trouble understanding, adding both options is recommended. However, there is often a correct and an incorrect form to consider.

Avoid aliasing conflicts:
Generally, the Josh.ai platform does not allow for the duplication of room and device aliases or scene trigger phrases. The purpose of doing so is to eliminate confusion when a user says, *"Ok Josh, turn on the chandelier"* when there are multiple chandeliers in a home. A recommended approach is to follow a *"<device name> <room name>" or "<room name> <device name>"* format, like *"Chandelier Living Room"* or *"Kitchen Chandelier"* respectively.

In large projects with many devices, long and specific names might be cropped due to app or UI character constraints. In these cases, the *"<device name> <room name>"* format is recommended to ensure that the specific device-type is visible while redundant room or area naming is the information being clipped. For example, *"Chandelier Living Room"* might be shortened to *"Chandelier Living..."* in Lutron's or Josh.ai's app, but the user is still able to clearly tell they are controlling the "*Chandelier"* in the Living Room.

A few instances to be cautious of are:

- Naming devices and rooms the same thing - Do not alias a light *"Living Room,"* instead name it: *"Living Room Pendant."*
    - This is especially problematic when thinking through what the off-state of a room should accomplish. A command like, *"Ok Josh, turn off the living room"* should control the various devices (lights, music, video, etc) in that room as opposed to a single device using the *"Living Room"* name.
- Naming devices in different rooms the same thing - The pendants across the Family Room and Theater should be named *"Family Room Pendant"* and *"Theater Pendant"* respectively.
    - A command to either room's *"Pendant"* light, will activate the corresponding device while creating the necessary naming distinction in Josh.ai's software.
- Naming a device its own type - If a particular light load is aliased as "*lights*" and the command is given *"Ok Josh, turn on the lights*" this device will turn on even if the intention was to turn on the lights in the room the command was given in. It is always best to avoid naming a device its own type, such as "*light*," "*shade*," "*thermostat*," etc.
    - Josh.ai will automatically attempt to prefix the device type with its room name if this conflict is detected.
- If a device or room is aliased the same as a scene, such as, "*Good Morning*" (it may sound odd, but it happens), and someone says "*Ok Josh, turn on Good Morning*" is the expected result to activate the device or the scene? For this reason, we want to avoid this sort of confusion whenever possible.

Add aliases:
When a user wants to refer to a room by multiple names, like "*Living Room*," "*Den*," and "*Family Room*," those aliases need to be added in Josh.ai. There are also different ways a microphone might hear a user depending on inflections or an accent. For example, *"Living Room"* might be misheard as "*Leaving Room*." By giving a few test commands, potential aliases will be captured in the chat

history and can be added to rooms or devices so that the client can refer to them in whatever way is most natural. Unlike physical buttons, which suffice with a single word or phrase, voice control requires some forethought into the various ways someone might want to refer to a room, device, or scene.

- Josh.ai attempts to proactively provide forethought whenever possible when it comes to room and device naming. For example, if a room is named *"Gym,"* Josh.ai will also accept a variation like, *"Ok Josh, turn on the lights in the fitness room*."
- Check the blue intuitive matches under room names in the Josh.ai web portal to see what common alternatives are automatically accepted.

Advanced Light Programming - Common Naming and Grouping Conventions:
Despite the emphasis on disallowing duplication of device naming in Josh.ai, it is possible to group devices together using the following unique naming conventions. For example, if you name one device *"Sconce"* that is located in the kitchen and another *"Sconce 2"* that is in the living room, saying *"Ok Josh, turn off the living room sconces"* will turn off *"Sconce 2."* However, using numbers in device names can become complicated quickly since the number *"2"* can be heard correctly as *"two"* as well as incorrectly as *"to"* and *"too."*

For this reason, it is a best practice to uniquely name each device with the *"<device name> <room name>"* or *"<room name> <device name>"* formats. Although two devices cannot be named *"Sconce,"* grouping will work as long as the keyword *"Sconce"* is in the light's name. A command like, *"Ok Josh, turn on the sconces everywhere,"* will activate every device that includes the label *"Sconce"* in the home.

The following light naming conventions are supported for uniquely identifying loads per room and grouping them in a multi-zone command:

- Light
- Sconce
- Chandelier
- Can
- Pendant
- Spot light
- Spot
- Torchiere
- Lamp
- Track

You can also precede the types above with any of the following keywords to form another group:

- Overhead
- Ceiling
- Hanging
- Floor

- Up
- Down
- Art
- Recessed

While some combinations are redundant *(Ceiling Chandelier)* or nonsensical *(Floor Sconce)*, this naming structure allows for references to devices like, *"Floor Lamps"* or *"Ceiling Lights"* or *"Up Lights"* or *"Down Lights."*

Some examples:

- *"Ok Josh, turn on the overhead lights"*
- *"Ok Josh, turn off the cans in the master bedroom"*
- *"Ok Josh, dim the down lights"*
- *"Ok Josh, set the floor lamps to 50%"*

Advanced Shade Programming - Common Naming and Grouping Conventions:
Similar to lights, there will be instances when common shade naming and grouping will be an important factor to consider when programming the optimal Josh.ai experience. When customizing the names for window coverings, there is lots of flexibility to offer a solution that will meet any user's needs.

The following window covering naming conventions are supported for uniquely identifying subtypes per room and grouping them in a multi-zone command:

- Shade
- Drape | Curtain *
- Blind
- Shutter
- Awning
- Blackout | Black out
- Sun | Solar | Sheer **
- Middle
- Center
- North
- South
- East
- West
- Northeast
- Northwest
- Southeast
- Southwest

* If a window covering includes the name *"Drape"* or *"Curtain,"* Josh.ai will allow you to refer to the window coverings by both names.
** If a window covering is named *"Sun," "Solar,"* or *"Sheer,"* Josh.ai will allow you to refer to the window covering by all three names.

Use Case 1: Single Uniform window covering on every window.

- Description of room: The client has a single automated shade/blind/curtain on each window.
- Naming: Since all the window coverings are similar, the suggested naming convention is for all devices with the same window covering type:
    - For example:
    - *"Breakfast Nook Shade"*
    - *"Doorway Shade"*
    - *"Window Sink Shade"*
- Commands: The client can give any of the following commands that will work:
    - *"Ok Josh, close the kitchen shades"* - All the window coverings close
    - *"Ok Josh, close the kitchen blinds"* - All the window coverings close
    - Note: They did not give the shades the alias of *"Blinds,"* but Josh.ai understands what shades the client wants to close

Use Case 2: Multiple window coverings on some or all of the windows

- Description of room: The client may have a blackout shade, sheer shade, blind, curtain, or any combination of window coverings on a window.
- Naming: Since the window coverings are different and the client may want to open or close a specific covering, we suggest naming them uniquely.
    - For example:
    - *" Breakfast Nook Blackout Shade "*
    - *"Breakfast Nook Sheer Shade "*
    - *" Breakfast Nook Curtain"*
    - *"Doorway Blackout Shade "*
    - *" Doorway Sheer Shade"*
    - *"Doorway Curtain"*
- Commands: The client can give any of the following commands
    - *"Ok Josh, close the blackout shades"* - Both of the blackout shades close
    - *"Ok Josh, close the sheers"* - Both of the sheer shades close
    - *"Ok Josh, close all of the shades"* - All four shades close (blackouts and sheers)
    - *"Ok Josh, close the curtains"* - Both of the curtains close

Use Case 3: Multiple walls of window coverings in the same room

- Description of room: The client may have window coverings on the east and west walls of the room.

- Naming: The client may want to open the East/West shades as a group, so those definitions need to be added to the device name.
    - For example:
    - *"West Blackout Shade Breakfast Nook"*
    - *"West Blackout Shade Doorway"*
- Commands: The client can give any of the following commands:
    - *"Ok Josh, close the West Shades"* - Both of the West shades will close.

Advanced Device Programming - Proximity Descriptors for Lights, Shades, and Fireplaces:
Depending on the location of a light, shade, or fireplace in a room, it can be named analogous to nearby decorations or furniture to provide additional contextual natural language capabilities. For example, if a light is named *"Portrait Light,"* then a user can tell Josh.ai to *"turn on the light near the portrait."*

The following descriptors support proximity-based understanding for lights, shades, and fireplaces in a room:
- Art
- Painting
- Portrait
- Tabletop
- Sink
- Island
- Cabinet
- Desk
- Door
- Bedside
- Vanity
- Couch
- Credenza
- Sofa
- Bench
- Counter
- Hutch
- Computer
- Mirror
- Piano
- Organ
- Harpsichord
- Fireplace
- Bay
- Headboard
- Trophy

- Bookshelf
- Bookcase
- Shelf
- Table
- (Pool | Billiards | Coffee) Table
- China Cabinet
- Chair
- (Rocker | Rocking) Chair

Although it is important to think through redundancies *(Fireplace Fireplace)*, this naming structure allows for even more natural references to devices like, *"Table Lamps," "Bedside Shades,"* or a *"Bookshelf Fireplace."*

Some examples:
- *"Ok Josh, dim the island light" = "Ok Josh, dim the light near the island"*
- *"Ok Josh, raise the cabinet shade" = "Ok Josh, raise the shade near the cabinet"*
- *"Ok Josh, turn on the bookshelf fireplace" = "Ok Josh, turn on the fireplace near the bookshelf"*

<u>CONTROL CAPABILITIES OF LUTRON DEVICES VIA JOSH.AI</u>

General Device Use:
- Josh.ai tracks the previously set level for dimmable lights and ceiling fans. When turned back on, each device returns to its previous level.
- If a device was discovered as the wrong type, it can typically be changed in the Josh.ai interface. For example, fireplaces can be assigned as such for a more accurate voice and UI experience.

Lights:
- On/off: Josh.ai interprets the "on" state of a light as the percentage or state that has been dictated in the lighting system's programming
  - *"Ok Josh, turn on the living room lights"*
- Brightness
  - *"Ok Josh, set the kitchen lights to 30%"*
- Conditional awareness
  - *"Ok Josh, it's too dark in here"*
  - *"Ok Josh, dim the lights a little"*

Ketra:
- Color:
  - *"Ok Josh, make the lights green"*
- Color Temperature:

    - *"Ok Josh, turn the bedroom lights to cool white"*
    - *"Ok Josh, make the lamp warmer"*
    - *"Ok Josh, set the lights to 3000 kelvin"*

Shades:

- Up/down:
    - *"Ok Josh, raise the drapes"*
    - *"Ok Josh, bring down the blinds"*
- Percentage adjustments:
    - *"Ok Josh, open the shades 50%"*
    - *"Ok Josh, bring up the sheers 25%"*
- Conditional awareness:
    - *"Ok Josh, lower the shades a little"*
    - *Ok Josh, elevate the curtains a lot"*

Thermostats:

- Temperature control:
    - *"Ok Josh, turn the kitchen thermostat to 72 degrees"*
    - *"Ok Josh, lower the temperature to 68 degrees"*
- Conditional awareness:
    - *"Ok Josh, I'm cold"*
    - *"Ok Josh, it's too hot in here"*

Fans:

- On/off
    - *"Ok Josh, turn off the fan"*
- Variable speed
    - *"Ok Josh, turn up the fan"*

Switches (Fireplaces, Garage Doors, Gates):

- Binary control:
    - *"Ok Josh, turn on the fireplace"*
    - *"Ok Josh, raise the garage door"*
    - *"Ok Josh, close the gate"*

<u>BENEFITS OF A VOICE CONTROLLED ENVIRONMENT</u>

The speed and precision of voice automation provides a friendlier way for users to operate their environment. Voice also offers a flexible alternative when searching for the correct wall-mounted switch or manually operating an app or touch panel are less efficient control options. Although more is going on in the background to process voice commands as opposed to a manual adjustment, the end-result is a simplified and "like magic" user experience. The ease-of-use for anyone to be able to converse with their technology is the next step in the evolution of user interfaces, eliminating the need to learn app flows or how to manually operate particular devices. Especially for users who are not technologically savvy, have visual or mobility impairments, or are seeking a way to future-proof their automation experience.

Voice control presents a compelling solution for a range of use-cases, primarily due to its efficiency offering an easier medium of communication. In a study by Stanford University, English dictation was proven to be three times faster than a state-of-the-art smart phone keyboard (Ruan, 2018). In addition to being a faster platform, voice proved to be more accurate in the same study with a 20.4% lower error rate when inputs were dictated instead of typed.

The recognition of the transformative power of voice control has resulted in an influx of smart speakers from prominent technology providers. Their advertising and mass market brand equity has not only created demand for smart assistants, but has opened the door to connected device adoption as well. While increased consumer interest presents opportunities for home technology professionals, it also becomes a challenge when trying to integrate disparate devices into an easily controllable solution. This is often the case because there is minimum viable compatibility between systems that are missing the full capabilities each device manufacturer offers within their own interface. The result is users jumping from app to app and needing to learn separate interfaces in order to control their environment to its fullest extent.

Josh.ai's concentration on home automation and deep compatibility with key partners like Lutron results in feature parity for all of its integrated devices. In doing so, Josh.ai's contextual natural language understanding and app as a traditional control option builds off of the familiarity of its partner's interfaces for users evolving their smart home. This approach effectively consolidates control under a single user interface, eliminates the app fatigue or clutter that is associated with most connected home ecosystems, and provides a more efficient and enjoyable experience.

When discussing the benefits of voice control, it is also necessary to touch on its negative connotations with respect to privacy and data security. Unlike the mass market providers that are strategically using smart speakers to collect and sell data about their users for marketing purposes, Josh.ai is only listening for its wake word. Once activated, Josh.ai then determines a command's legitimacy relative to home automation or a valid personal assistant feature. If Josh.ai determines that it was woken up accidentally due to a side conversation or television dialogue, then it does not log that input as a command and stops itself from falsely verbally responding. With a home-focused approach, Josh.ai does everything in its power to ensure that what is said behind closed doors stays there and that it is mindful of unsolicited interruptions.

Voice has already made significant inroads into the smart home space due to its convenience. However, the Covid-19 pandemic presents an opportunity for home technology professionals to provide it as a safety measure for clients as well. As homeowners look for ways to avoid commonly touched surfaces like tablets, remotes, light switches, and more, voice will be a key driver in extending the range and adoption of smart home technology (Carlaw, 2020). Lutron's QSX product line offering circadian rhythm lighting will be one such technology that stands to benefit. Voice will position itself as a trojan horse for broader wellness education, which will enlighten users to the benefits of tunable lighting that naturally signals transitions throughout their new home-based daily routines.

CONCLUSION

Voice control's evolution from a novelty to an expectation in many projects makes it more critical than ever for professionals to have a blueprint to follow for successful deployments. Success will not only be based on following best programming practices, but also on the system's functionality being designed for optimal verbal interaction. By taking the approach of utilizing voice's efficiency in combination with a manual interface for more granular control or as a backup option, installers will ensure projects have the mechanisms in place to provide innovation with reliability.

The residential space has been the primary beneficiary so far and there is plenty of opportunity on the horizon. Going forward, any living space or automated environment should be positioned for voice control in order to decrease friction between technology and its users.

RESOURCES

1. Ruan, S. (2018, January 17). Speech Is 3x Faster than Typing for English and Mandarin Text Entry on Mobile Devices. Retrieved July 27, 2020, from https://hci.stanford.edu/research/speech/paper.html
2. Carlaw, S. (2020). TAKING STOCK OF COVID-19: The Short- and Long-Term Ramifications on Technology and End Markets. *ABI Research*. Retrieved July 27, 2020, from https://cdn2.hubspot.net/hubfs/6705264/Marketing/Whitepapers/Taking%20Stock%20of%20COVID-19/ABI_Research_Taking_Stock_Of_COVID-19.pdf?hsCtaTracking=f4acd515-e503-4d19-bdbf-59f060ff307f%7C81549895-e101-4dd7-85d2-1edfa5ba95ea