
# JubiAudio Wireframe Design

**Rev 2016-1118_D**

Steven Wang *CEO*

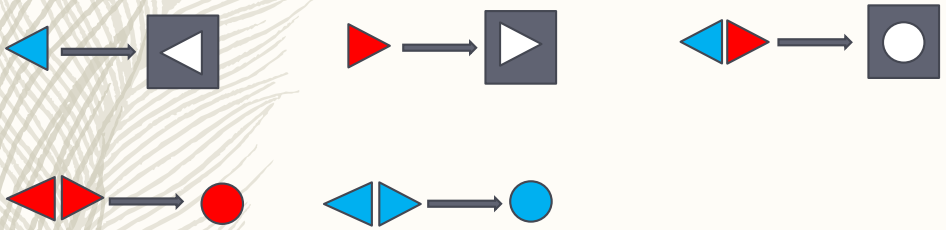
Confidential for internal distribution

# Phase I

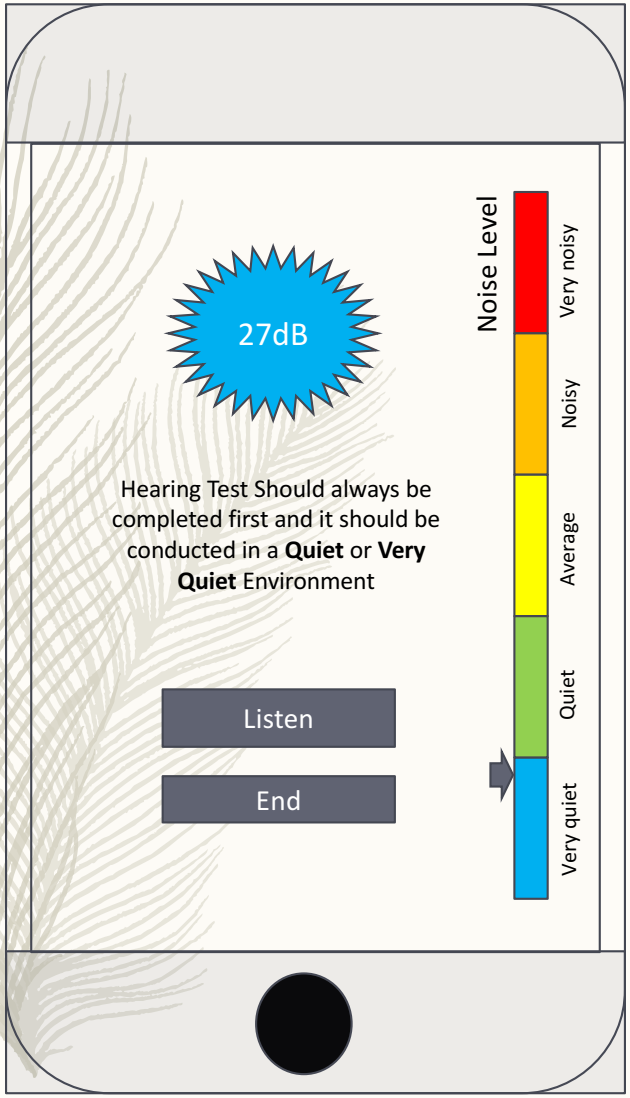
Two wireframes.

- Wireframe-01 is to demonstrate the ability to capture and display the surrounding audio signals, which is interpreted as the “noise” of the environment.
- Wireframe-02 is to demonstrate the ability to
  - Develop the App as closely and feasibly as possible using the MVC model.
  - Create an interactive equalizer
  - Plot a static curve and superimpose on top of the equalizer properly
  - Allow the input audio samples to be fed to the JubiEngine and strictly to the microphone

Emoji suggestion, if art works are not available, use the following Emojis as buttons



## Noise Level Detection
(Wireframe-01)




This is the initial screen, i.e **home screen (HS)**, when the App is turned on or the [**home**] button is pressed. The microphone(s) will pick up the surrounding noise and display the noise in dB (1) in the center with proper color and (2) move the pointer along the Noise Level Bar in the correct color region and scale. The colors of the noise level do have meanings, therefore, it is preferred to use a color system close to this wireframe.

iPhone comes with 16 discrete volume controls. We advise to use the lowest level as 0 dB, which will be used as the reference to determine the dB for all the other levels. Note: by iPhone default, a return value of 0 dB indicates full scale, or maximum power; a return value of -160 dB indicates minimum power (that is, near silence). see: https://developer.apple.com/reference/avfoundation/avaudiorecorder/1387176-averagepower

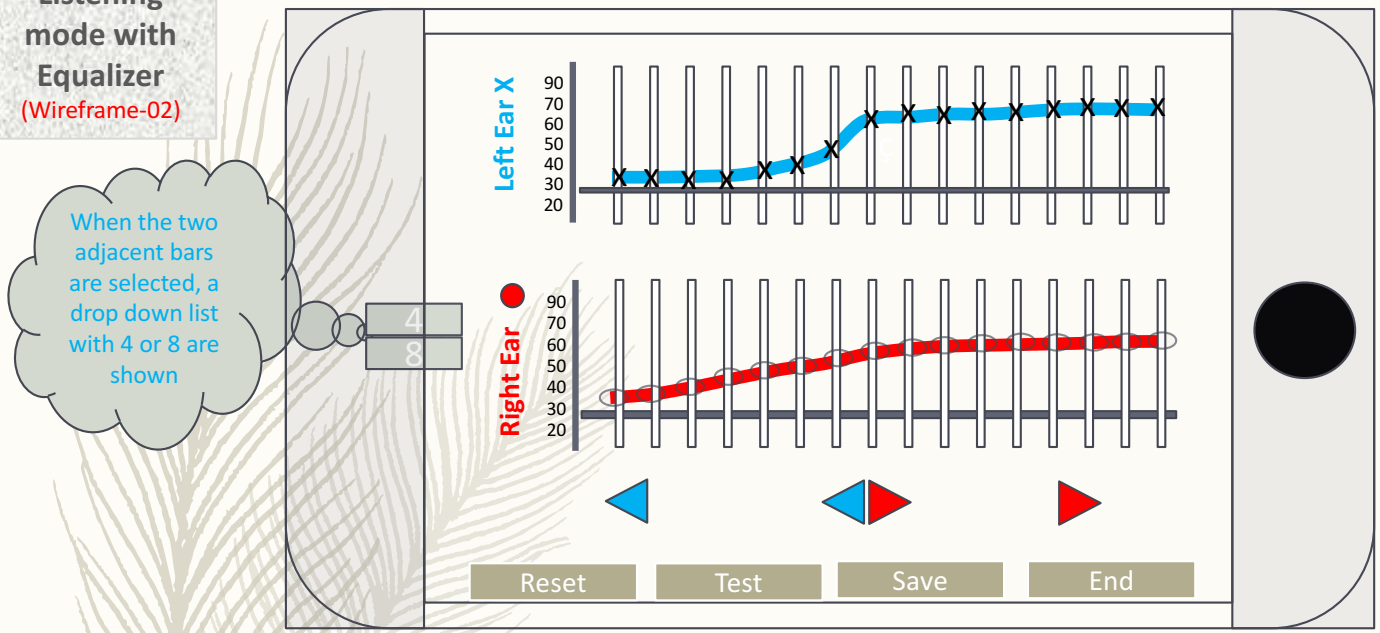
The UI program is preferred to work with a managing program, Manager, to receive, interpret and display. When the user presses the [**Listen**] button, the noise dB number should be recorded into the user's profile database (ProfileDB) which may be in the Manager as a record: {UserName, UserEmail, UserPhone, NoiseLevel, Audiogram, EQRSetting}. At this moment only the NoiseLevel is recorded. All other fields are reserved for the future usage.

Audiogram is an array of M tuples: (Frequency, dB). M = 16 as default. This ProfileDB shall be stored within the iPhone and be retrievable in the future.

EQRSetting is a double-array of N tuples of {Frequeny, dB), one array is for left equalizer (EQR) and the other for right EQR. N is from 16 to 64, typically.

- [**Listen**] button activates Wafeframe-02 and start to listen using the JubiAudio engine.
- [**End**] go to **homescreen**

## Listening mode with Equalizer
(Wireframe-02)




The real-time audio wave is moving up and down with the static band tube

Go to left-mono EQR screen. Only process left audio input & feed to left output

Go to right-mono EQR screen. Only process right audio input & feed to right output

Display both sides EQR and process in stereo mode

and can be replaced by the standard iPhone emojis

Note: Fine-grained band expansion can be achieved in the single ear listening screens explained in the later wireframes.

This is the default screen for **Stereo-mode** Listening/Equalizer. The microphones are activated and the audio are passed to JubiAudio engine (JubiEngine) for processing.

If the audiograms contain values, they should be displayed as static Blue and Red curves behind the equalizer, and the EQR knobs are placed at the right gain positions as indicated in this wireframe. Otherwise, no Blue and Red curves should be displayed and knobs should be placed at the baseline, say **20 dB**. The discrete frequency bands of EQR are displayed as static hollow tubes and dynamic moving waves within.

The [X] and [O] are the adjustable knobs of the equalizer (**EQR**). User can adjust the EQR gain for each of the frequency band by moving up and down the knob. Real-time positions of the EQR knobs shall be passed to the Manager then to the Engine to change the gains in real-time or near-real-time.

- [**End**] terminate the program & go to home screen
- [**Test**]– Both audio inputs from the microphones will be processed by the App's engine but only the unprocessed input will be fed to the headset directly
- [**Save**] - the App records the current EQR settings in EQRSetting for future use
- [**Reset**] put all the knobs to the base line