# Ideas

## Virtual Guitar Assistant

A sequencer that controls a number of tracks that could contains audio or midi events. The sequencer’s tempo is controlled by an algorithm that tries to keep up with a real-time guitar source.

### Juce POCs

## Patches Guitar Dashboard

* A Device Patch can be defined by a device, a program number and a scene number
* A Song Patch can contain more than one Device Path and have a name
* A Marker is e time point in a track
* A Track contain an audio file that can be played at various tempos and also can contain a list of Markers.
* A Song is a collection of Song Patches, a name, a track
* Songs can be collected in Song Collections
* Song Collections can be collected in Libraries.

### Juce POCs

* Receiving Keyboards events.
* Receiving e sending midi messages

# Tutorials

Playing a sound from file.

Call AudioFormatReader\* AudioFormatManager::createReaderFor (const File& file)

auto newSource = std::make\_unique<juce::AudioFormatReaderSource> (reader, true);

AudioTransportSource::setSource (PositionableAudioSource\* const newSource,

int readAheadSize, TimeSliceThread\* readAheadThread, double sourceSampleRateToCorrectFor, int maxNumChannels)

AudioTransportSource::start()

# BackLog

* ~~After loading the ui must be updated~~
* ~~Full screen at startup~~
* ~~Default loading at startup~~
* ~~VirtualBand component on screen at startup~~
* ~~After selecting a song the program changes component must be updated.~~
* ~~Selecting a program change must send program change messages~~
* ~~Previous next marker implementation~~
* ~~Current song/program change must be highlighted~~
* ~~Program change / scene optimization~~
* Scene optimization based on the patch default scene
* Choose a program icon
* Patch number and scene as tooltip on ProgramChangesComponent
* Program number verification
* Turning on and off the devices must be detected.
* The Audio/Midi setup must be saved and restored
* ~~Add a slide to PlayerComponent to track the song position during play~~
* ~~Add a visual reference for every marker in the song~~
* ~~Implement the ‘next marker’ and ‘previous marker’ button in PlayerComponent.~~
* Implement the Song::deactivate by stopping the playing track
* ~~Loading the correct track for every Song object~~
* ~~The tile’s background colors must change according to tile’s ordinal number.~~
* ~~Restore the selected tile look~~
* ~~The ProgramChangeComponent should display its tiles in a single line. The selected program’s tile should be at the center (if possible) and the switching between one program to another should be animated.~~
* ~~The ‘b’ and ‘space’ key should control track playng~~
* ~~Holding ‘b’ or ‘space’ key should rewind the track~~
* ~~‘Right’, ‘Shift/Right’, ‘Left’ and ‘Shift/Left’ keys should control track position while playing~~
* Associate every program change to a position within the song so that during the song’s track playback the program change can be automatically played back, if the function is enabled. So there must be a sort of toolbar for enabling/disabling the function
* ~~The selected song/program\_change must be more visible: thicker border and lighter tale background.~~

# Bugs

* ~~PlayerComponent buttons does not respond to the setEnable call.~~
* ~~FractalDevice::loadAvailableDevices must be synchronized with VirtualBand::loadSongLibrary~~
* ~~Slider should display the track position as HH:MM:SS.mmm~~
* ~~Moving cursor on the slider should change track position.~~
* ~~Fix the memory leak shown at program exit~~
* Songs must be playable on both Windows and MACos
* ~~Marker must be rendered properly~~
* Next and previous marker commands should work also when the song in paused.
* Song position should return to zero when the song reaches the end.

# Improvements

* Every component could have a background image like the one used by web.whatsapp
* Patch and scene numbers should be visible in the status bar

# Testing

* ~~The new setup audio page~~