# 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
* The tile’s background colors must change according to tile’s ordinal number.