**Capstone Weekly Project Summary**

Keep your total weekly project summary to a single double-sided printed page.

|  |  |
| --- | --- |
| Week 1 | Project Status: N/A (initial meeting) |
| Tasks Completed/New Functionality | * Pre-Capstone, a some research and prototyping done * Java determined to be the language for implementation after C# was found to be not feasible |
| Comments | Challenges to overcome:  -FFT implementation was difficult and had too much inaccuracy –autocorrelators found to be accurate, project will use YIN autocorrelator. |

|  |  |
| --- | --- |
| Week 2 | Project Status: Green |
| Tasks Completed/New Functionality | * Added Western Scale (all notes from 55 Hz to 1000 Hz, range of human voice and classical guitars) * Visual feedback of note singing – note is displayed when sung on pitch with visual cue “ON PITCH”. * Removed unneeded components from vendor code. * Added implementation of GUI for the multi-part app. * Research, prototyping GUIs, adding western scale, finishing note detection, 14hrs work. |
| Comments | Challenges to overcome:  -How do I deal with vibrato?  -What constitutes a pitch sung “on key” – whole duration average (problems)? in chunks?  Next Week: Start/Stop sampling on correct panel, vibrato problem solved, final pitch determining, scoring.  If time, begin perfect pitch implementation. |

|  |  |
| --- | --- |
| Week 3 | Project Status: Yellow |
| Tasks Completed/New Functionality | * Consensus is that perceived pitch “correctness” is based on the high and low vibrato pitch being the SAME amount above and below the correct pitch. This is not a linear amount due to the wider spacing of pitch as notes go up the scale. * Implemented the first app on the correct tab, determined vibrato benchmarks,   gives feedback on average pitch frequency (approx. 30 samples per sec).  Research, coding, debugging, 13 hours work. |
| Comments | Challenges to overcome:  -There are a lot of overtones, something that is a weakness of autocorrelators. I need to create a filter to get rid of these.  -Right now the average accuracy is good on low notes but we lose accuracy on the high due to the spacing between frequencies of pitches. Find a way to account for that.  Next week-  -get the pitch average accuracy up for high notes.  -create filter for overtones  -complete the pitch play /sing back app |

|  |  |
| --- | --- |
| Week 4 | Project Status: Red |
| Tasks Completed/New Functionality | * Filter created to get rid of over/undertones * Pitch play/sing back part of app progress- tone generator |
| Comments | Challenges to overcome: The application is getting messy and unorganized – I need to refactor and make it more in line with the MVC structure so it is more easily changeable.  Next week-  -Refactor code into MVC structure |

|  |  |
| --- | --- |
| Week 5 | Project Status: Red |
| Tasks Completed/New Functionality | * Refactor entire code base to reflect an MVC structure for maintainability and extensibility * Bug fixes |
| Comments | Challenges to overcome:  Next week- (The hardest, largest part of the app)  PART 1  -Finish previous  -Implement persistence for short musical phrase including pitch assessment  -Record sung musical phrase, do file analysis on it |

|  |  |
| --- | --- |
| Week 6 | Project Status: Red |
| Tasks Completed/New Functionality |  |
| Comments | Challenges to overcome:  Next week- (The hardest, largest part of the app)  PART 2  -Implement persistence for short musical phrase including pitch assessment  -Store performance +analysis data – create report file to print out / display |

|  |  |
| --- | --- |
| Week 7 | Project Status: Red |
| Tasks Completed/New Functionality |  |
| Comments | Challenges to overcome:  Next week- (The hardest, largest part of the app)  PART 3  -Implement persistence for short musical phrase including pitch assessment  -Visually represent |

|  |  |
| --- | --- |
| Week 8 | Project Status: Red |
| Tasks Completed/New Functionality | * more MVC refactoring * BUG FIXES * GUI layout designed, implementation begun |
| Comments | Challenges to overcome:  -Java guis are complicated and time consuming to construct, even harder to make them look good.  -Difficult getting all the components to talk to each other – takeaway: design and implement MVC from the beginning, not a re-factor nightmare.  Next Week-  -MVC finished  -All apps finished except Session Training  -Analyzer Tests |

|  |  |
| --- | --- |
| Week 9 | Project Status: |
| Tasks Completed/New Functionality |  |
| Comments | Challenges to overcome: |

|  |  |
| --- | --- |
| Week 10 | Project Status: N/A (presentation week) |
| Tasks Completed/New Functionality |  |
| Comments |  |

Project Backlog-

Features already planned for the project, from research and SME:

1.) A vocal student or musician would like to determine how accurate their ear is during warm-up exercises. They will sing or play a pitch and the system will give feedback on its accuracy based on its closest frequency.

2.) A vocal student is training to have perfect pitch, meaning they would like to envision a note in their mind and sing it accurately without any auditory prompting. The student enters how many random notes to try and the range of those notes. The system presents the name of a note to the student; the student sings the note and receives accuracy feedback based on the correct frequency.

3.) A student would like to see a visual representation of their performance as it is in progress. As the student varies the pitch of their note, the system provides a visual representation of pitch performance.

4.) A music instructor would like to evaluate a student’s pitch performance on a musical phrase or song. The system captures the student’s performance; it analyzes the performed music and displays accuracy feedback for the instructor and student based on closest frequency comparisons. The instructor saves the session for further review.

Added after project start:

1.) Computer plays a note, singer sings back the note, accuracy feedback given.

2.) Computer plays a note, singer identifies it.

3.) Interval training- play a random note and have the student sing a given interval from that note, for example, a fifth. Evaluate the student's pitch.

4.) Chord progression training for instrumental, as in for a guitar.

5.) Port to IPhone / Android / Windows Phone