- Dec. 15 - Jan. 4
    - Added a new property to the current Lesson Data structure in order to determine length of a note value
    - Added the ability to program “Rest” values in into lessons
    - Able to compare user’s input with the lesson input accurately
    - Started working out how scoring of the attempts will go, still need to test out more
- Jan. 4 - Jan. 10
    - Continue to work on scoring algorithm in order to properly give users a score
    - ensure scoring threads have mutual exclusion in order to allow the score to be incremented properly
- Jan. 11 - Jan. 17
    - Create a parsing algorithm in order to retrieve lesson data from a text file and store the values to the corresponding variables
    - Work on a easy-to-use structure for the lesson text file, so that programming new lessons is easy.
- Jan. 18 - Jan. 24
    - Create an algorithm to take user input and save it to a readable text file, so it can be later output to the user
    - Create an algorithm to take the lesson data and convert it to a readable text file, so it can be previewed by the user before and after the lesson attempt
- Jan. 25 - Jan. 31
    - Create the lesson feedback data structure, with the necessary getter and setter functions
    - Create a comparison algorithm which uses the readable text files generated in order to produce accurate feedback for a user after the lesson has been completed
- Feb. 1 - Feb. 14
    - Format a text file, which may be read and modified, to store important information and statistics about the user’s progress.
    - Look into JSON or XML formats to store user and lesson data
    - Create a ‘user’ class object which may be created and stored in a ‘boot’ file so multiple users may be created and when the program is loaded they may select which user is playing. The boot file should contain the information necessary to create an instance with the proper information, access the correct files with the user’s statistics, and have the proper values for unlocked lessons.
- Feb. 15 - Feb. 29
    - Ensure program login, lesson preview, user attempt, and lesson feedback components are properly working together by generating files and data under multiple temp users. Then logging out and logging back in to ensure the correct lessons and data are available.
    - Optimize the critical components in order to have the best speed and highest accuracy. Accuracy is more important than speed here but some components, such as file access, should have the highest priority set for speed.

- March 1 - March 31
    - Reorganize the entire project to have the correct flow through screens.
    - Separate components into different class, interface, and abstract objects in order to increase the abstraction of the program. Also, this will ensure the program is more reliable so that if one component is experiencing issues, the other components will not suffer.
    - Distribute the application to a set of, both, musical and nonmusical individuals in order to test the software and report back problems and results.
- April 1
    - Work on making the program more user friendly.
        - Drop down menus rather than text input
        - Remove any excessive buttons, text fields, and other I/O components that are not necessary to the user's experience
        - Make the GUI more attractive
        - Package the program in the most light-weight way possible so the user does not have to sacrifice an enormous space of memory to install
        - Make the install process easy to use for the user, as well as the uninstall process
    - Create a way for the program to read "Update" files so the user may add, or create, new lessons and features to be used by the program without having to uninstall the current version to re-install the latest version.
    - Ensure the program boot files are protected and secure so that the program may not be corrupted.
    - Again, distribute the application to a set of, both, musical and nonmusical individuals in order to test the software and report back problems and results.