App 1: Pitch Perfect

Criteria	Does Not Meet Specifications	Meets Specifications	Exceeds Specifications (Completely Udacious)
Basic Functionality			
The app contains two scenes of content: one for recording an audio file, and one for playing the audio with different effects.	The app does not have the basic structure specified.	The app contains the two pages of content (for recording and playing audio) and uses UINavigationController to navigate between these two scenes.	*Not Applicable*
All UI elements (buttons and text) are appropriately formatted for iPhone portrait layouts.	UI elements are not properly positioned on the screen for iPhone portrait layouts.	UI elements are appropriately positioned on the screen for iPhone portrait layouts.	*Not Applicable*
Actions and Outlets			
The app uses IBAction methods to record audio and playback sounds.	The app does not successfully connect buttons in the storyboard to actions in the code.	The app correctly connects each button on the storyboard to its own IBAction method.	*Not Applicable*
Labels and buttons are shown or hidden as appropriate.	The "Recording" label or the "Stop" button are visible even when recording is not in progress, or do not appear properly when recording is in progress.	The "Recording" label and the "Stop" button are hidden and shown appropriately: they do not appear by default, but correctly appear when recording is in progress.	*Not Applicable*
AVAudioRecorder			
The first scene of the app uses AVAudioRecorder to record audio.	The app does not record audio successfully.	The app successfully uses AVAudioRecorder to record audio.	*Not Applicable*
The app keeps track of the recording using a custom model class.	A custom Model class is not used to save recorded audio.	A custom Model class is used to save recorded audio.	*Not Applicable*
Delegates and Segues			
The app uses the audioRecorderDidFinishRecording function to determine when the audio has finished recording.	The app does not use audioRecorderDidFinishRecording().	The app uses the delegate pattern and implements the audioRecorderDidFinishRecording() method.	*Not Applicable*
The app programmatically triggers a segue from the first scene to the second by using the performSegueWithIdentifier function.	The app uses a hardcoded Storyboard segue.	The app does not use a hardcoded Storyboard segue. A segue from the first scene to the second is programmatically triggered via performSegueWithIdentifier().	*Not Applicable*
UINavigationController			

The app allows users to re-record audio after a recording is complete.	The app does not allow re-recording. Once the recording is complete the sound is fixed.	The app allows the user to re-record by navigating back to the first scene from the second.	The app allows users to pause and resume recording.
Sound Effects			
The second scene of the app contains the following audio effects: Snail (slow), Rabbit (fast), Chipmunk (high pitch), and Darth Vader (low pitch).	Not all four buttons are present; or the buttons are present, but the app doesn't play the sounds associated with the buttons.	The second scene of the app contains the following buttons for audio effects: Snail (slow), Rabbit (fast), Chipmunk (high pitch), and Darth Vader (low pitch). All four buttons work properly to play the associated effects.	The app showcases at least one additional audio effect, such as echo or reverb.
Code Improvements	This Code Improvement document lists four tasks that you must complete on your own to improve the code written in this class. Your final code must implement all four of these tasks.		
Task 1: The model class uses an initializer, and this initializer is called in RecordSoundsViewController.	The model class does not use an initializer, or the initializer is not properly called in RecordSoundsViewController.	The model class uses an initializer and this initializer is called in RecordSoundsViewController.	*Not Applicable*
Task 2: The bug where sound effects overlap during playback is removed.	Sound effects overlap during playback.	The bug where sound effects overlap during playback is removed.	*Not Applicable*
Task 3: Legacy, commented-out "dead" code is removed from the project.	Legacy, commented-out "dead" code remains in the project.	Legacy, commented-out "dead" code is deleted from the project.	*Not Applicable*
Task 4: Meaningful information, such as a Tap to Record button, is provided to guide the end-user.	The label "Tap to Record" does not appear before the microphone icon is pressed, or does not change to "Recording …" when recording is in progress.	The label "Tap to Record" appears before the microphone icon is pressed, and changes to "Recording …" when recording is in progress.	*Not Applicable*
Code Quality			
Code is effectively abstracted.	Code is unnecessarily repetitive and not effectively abstracted into reusable methods.	Potentially repetitive blocks of code are effectively abstracted into reusable methods.	*Not Applicable*
Code adheres to Swift naming and style conventions.	Code does not adhere to Swift naming and style conventions.	Code adheres to Swift naming and style conventions.	*Not Applicable*
Code uses appropriate and effective comments.	The code has hard-to-follow blocks that are not well-commented.	Code is readable and easy to follow. Any code that may be hard to understand is commented effectively.	*Not Applicable*