# Sound Design for Games

# Assignment 2 Tech Log

**Elvis Suhadolnik Bonesso**

# Table Of Contents

# Introduction

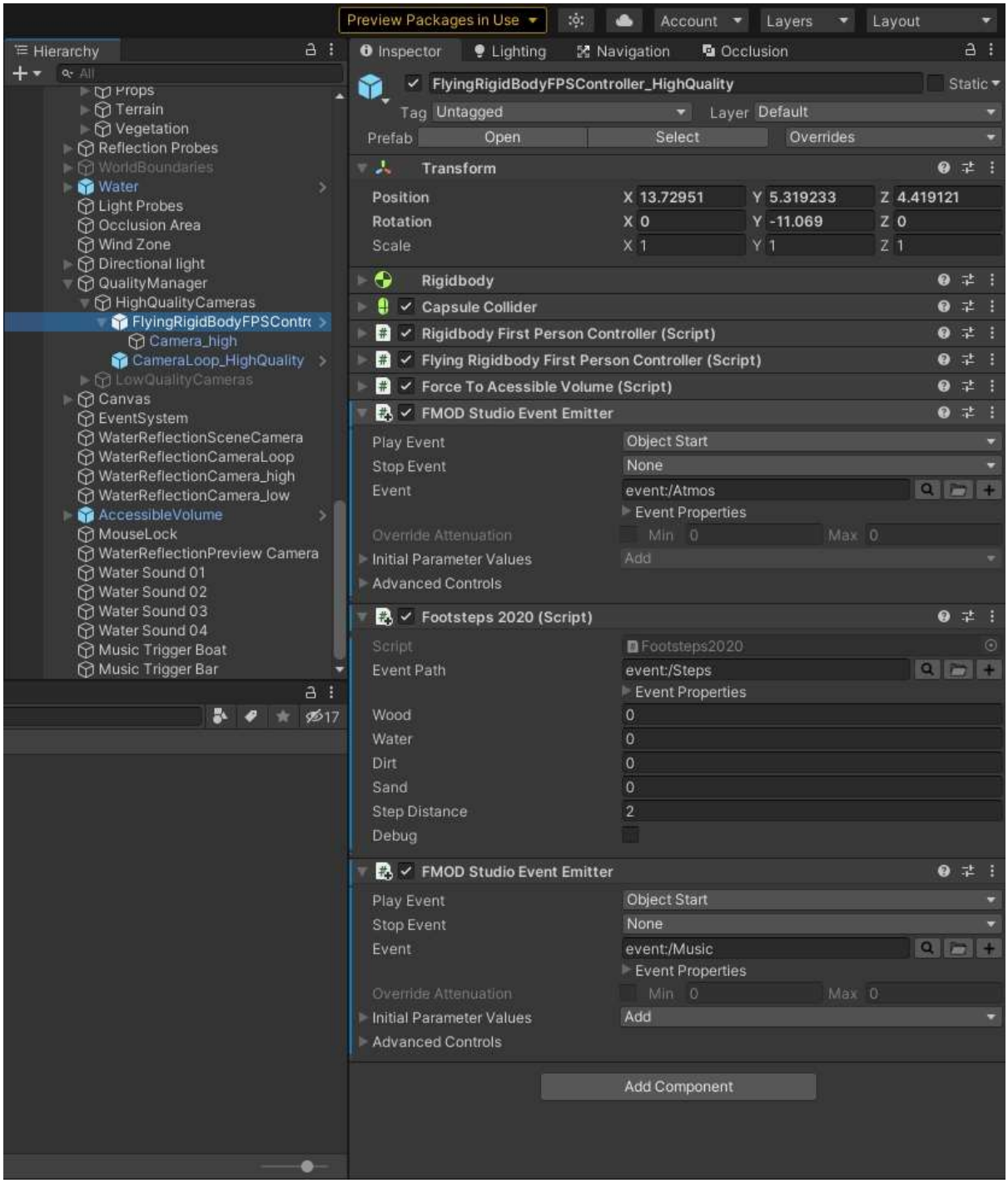
This technical log intends to describe the workflow and creative processes used to complete the Sound Design for Games assignment. Students were asked to create and implement all sounds for the Viking Village from atmospheres, crowds, and torches to steps and music implementation, using FMOD and Unity.

# Process

Starting from playing the game in Unity, a list of things would need to have implemented sounds was made, stating the following:

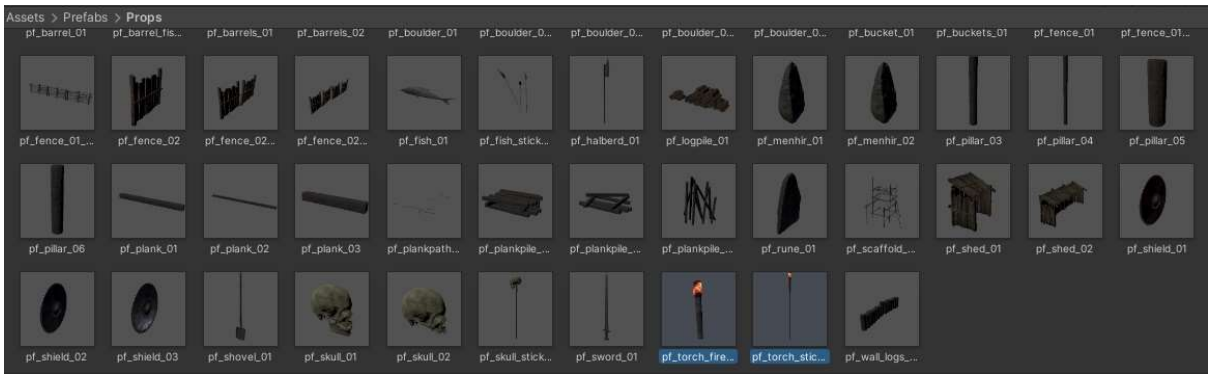
Atmosphere
Steps
Torches
Water
Wheel
Barn
Workshop
Skulls
Bar
Boat


The first step was to create an atmosphere with insects and birds. In FMOD, three audio layers were added with sounds from different outdoor environments, balancing the overall volume and placing a loopable section so the track 3 loops seamlessly. For some variety in the overall sound, the clip in track 1 was within a multi-instrument set to async, this way it'll hardly loop together with track 3. Track 2 had a probability on in around 50% with a fade out by the end of it, in case the track goes off. Volumes were cautiously balanced so the difference goes unnoticed if track 2 goes on in one of the loops. In Unity, this was attached to the *FlyingRigidBodyFpsController* as an event emitter, so the event always plays as the character goes anywhere in the game.



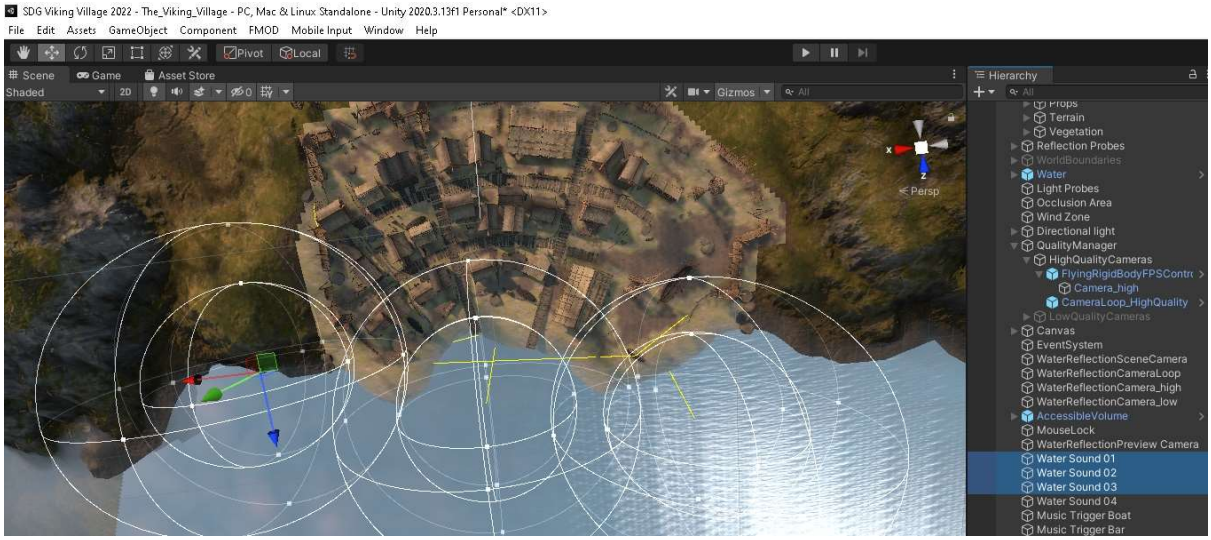

There were four different grounds the player could step on: Dirt, Sand, Water and Wood. Four tracks were made for each of these grounds and within them a multi-instrument was added with four varying steps for each ground, set to 25% of chance to be played in each. In order to make these steps work with the provided script in Unity, four parameter sheets were added with the name of these different grounds and within them, the tracks had their volume automated, increasing the volume up for the named parameter and decreasing the others to zero, respectively. This way, if the characters steps in the dirt, the script will trigger the dirt sound and so on. In Unity, this was attached to the *FlyingRigidBodyFpsController* as a Footstep script, provided by the teacher. The script also had to be placed within the assets > scripts.


There were two types of torches, the big and the small ones. Using the same audio files in different ways it was possible to create a tone difference between them. For the big ones (torches on a stick) the second track (the one with the stronger sound) was louder, and part of its clip was trimmed to be used on a loop within a multi-instrument with the async feature turned on. For the small torches, the process was the exact same but this time the second track was quieter, this way the first track had the most prominent sound, setting the tone for those torches. In Unity, in order to attach the sounds to all respective torches at once, the student place a FMOD event emitter in the torches that are in the assets > prefabs > props.

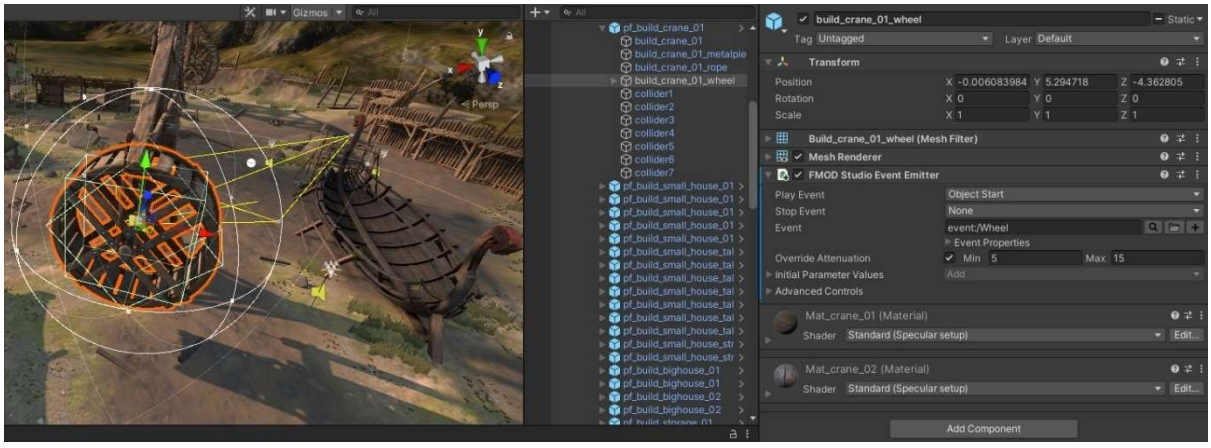



For the water, two tracks were added. The first one was a really long audio of the sea, with some wind, the sound of the shoreline and a few waves. This track was added to a multi-instrument, set to async, and the second track, which were some wave sounds, was set to -12db as the sounds can get really noticeable if they were louder than that. First track also needed some equalization to cut off the low end and some of the high end. In Unity, some

spheres were created and placed in certain sections of the shore with their mesh filter and sphere collider turned off, adding also an event emitter linked to the *Water* event.

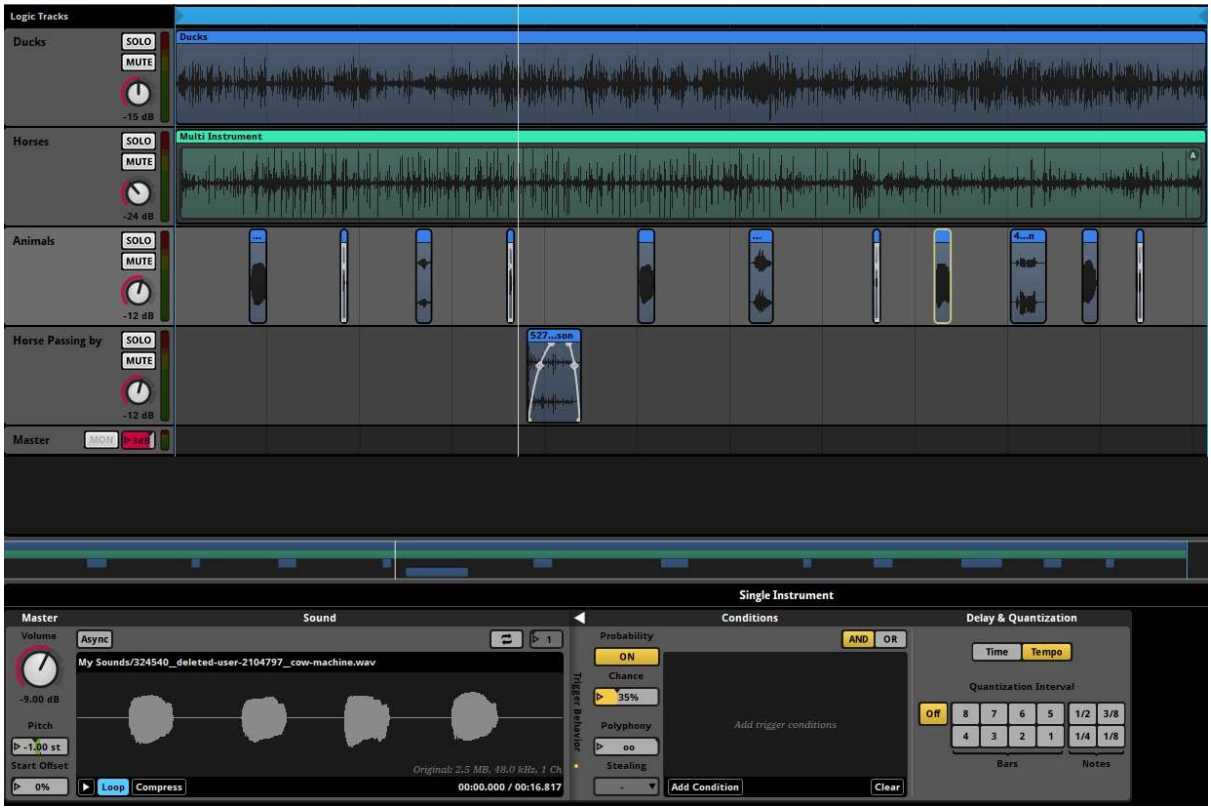



For the big wheels, a slowed varisped wooden sound was added to the first track, looping with the second one that had an asynched multi-instrument with some metal creaks. The other two tracks had many other metal and wood creaks with their probability set to 30 and 75% respectively. In Unity, an event emitter was added to both of these wheels with the override attenuation set between 5 and 15.

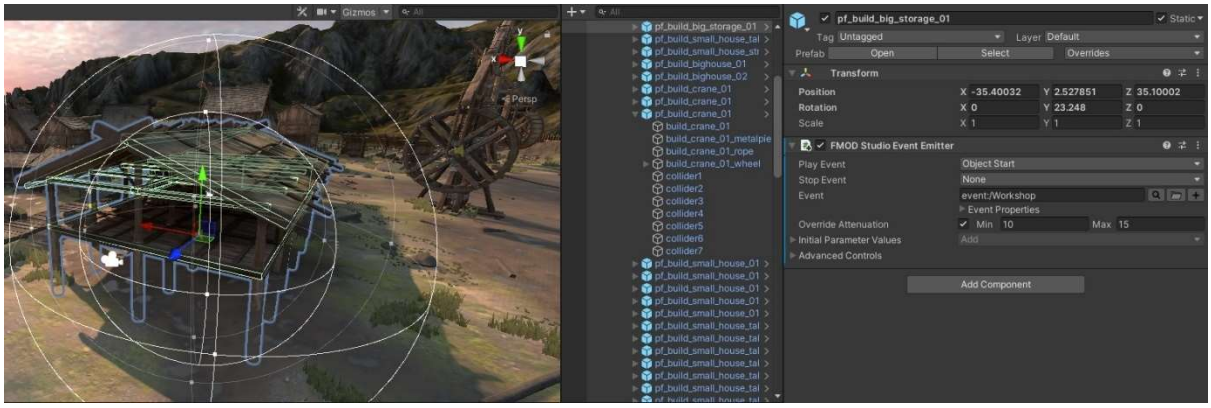



For the Barn, there were two main tracks playing in a loop (one as a multi-instrument set to async and the other one just looping normally). The first track had some ducks and other small animals and the second one some steps of big animals, such as horses and cows. The third track was made to use sounds of many different animals such as horses, cows, rooster, among other animals. These audio clips had their probability set between 30-70%. And the last track was the audio of a horse passing by with its probability to play set to 25%. In Unity,

an event emitter was added in the place with the override attenuation set between 13 and 20.




The workshop had a similar structure, two base tracks, one of someone hitting stones within a multi-instrument set to async on loop and another track of someone doing some woodwork cut in two sections but with its probability set to 75 and 85%. All the following tracks are composed of “different workers” getting their work done in similar ways using clips with their probability set randomly from 50% above. In Unity, an event emitter was added the place with the override attenuation set between 10 and 15.



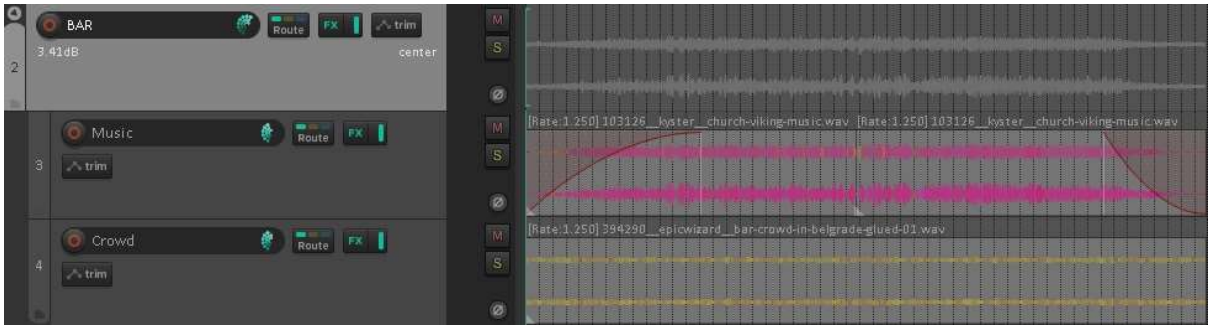
The Skull idea came on later but there were so many skulls around the game that making a sound for them was quite an invitation. The idea was to give dead people a voice, whispers were the first thing that came to mind and it fits them quite well. In FMOD, a loopable section was added, so a high pass filter was used in the track and to cut off the low end and later the track was sent to a delay unit. Another track was added with same settings apart from the volume set to -9bd, using the reversed version of the same audio file. In the master channel of this event, a multiband EQ was added taking out the low and high end as well taking out some of the frequencies in around 2k as it was getting muddy. In Unity, in order to attach the sounds to all respective skulls at once, the *FMOD event emitters* were placed in the skulls that are in the assets > prefabs > props.


For the bar and the boat, it was decided to include all the audios on the same event called music, as all these places would influence the main music playing in the background.

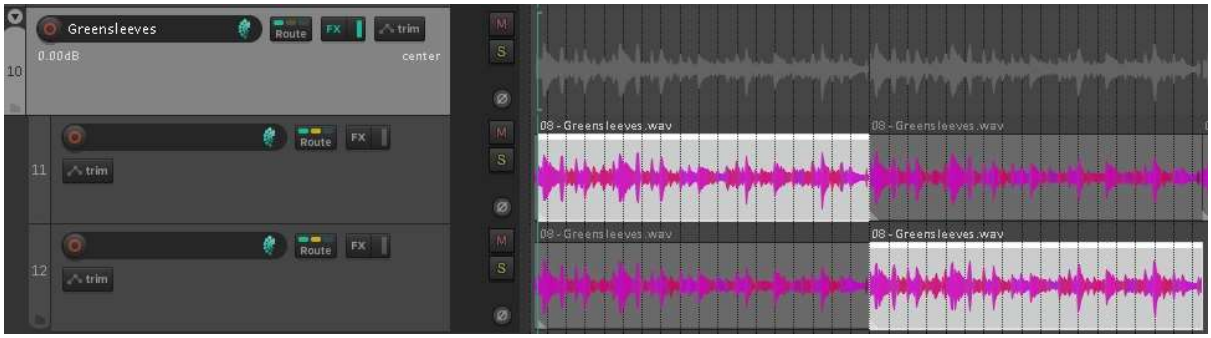
For the boat, the student decided to create an inviting atmosphere as if Odin would be calling the player to go onwards with their discovery. To make this possible a choir from

Native Instruments, *Mysteria*, was added. To record the audio the student used the abstract chord preset, playing around with its density and intensity, later coming back to the same starting point to make the looping possible.

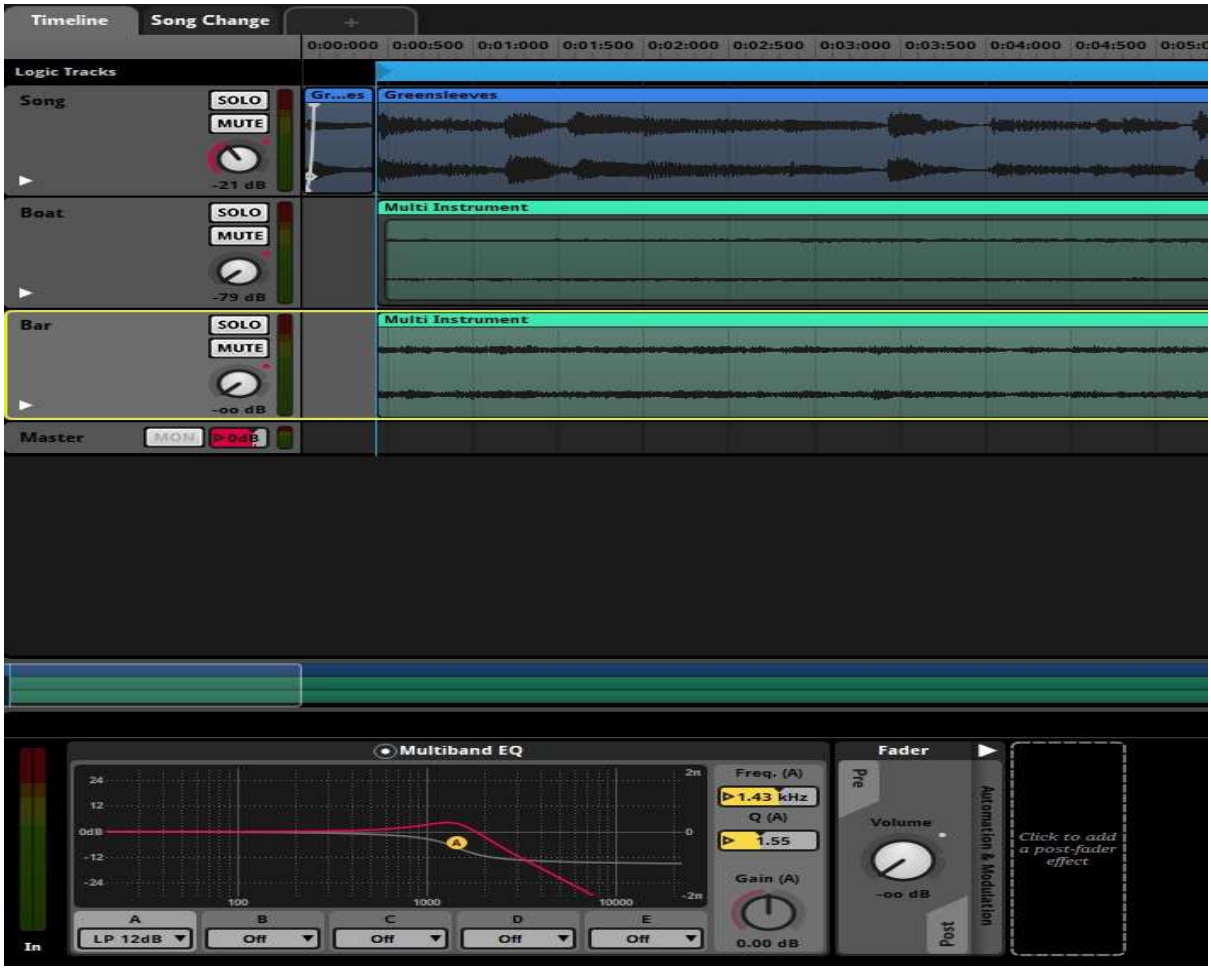

For the bar it was imagined that a crowd would be there talking loudly while the musicians are playing so two audios were used to create this section: a crowd in a bar and musicians playing Viking songs around a small crowd. The songs were slowly turned down and up again while the crowd keeps talking in a loop.



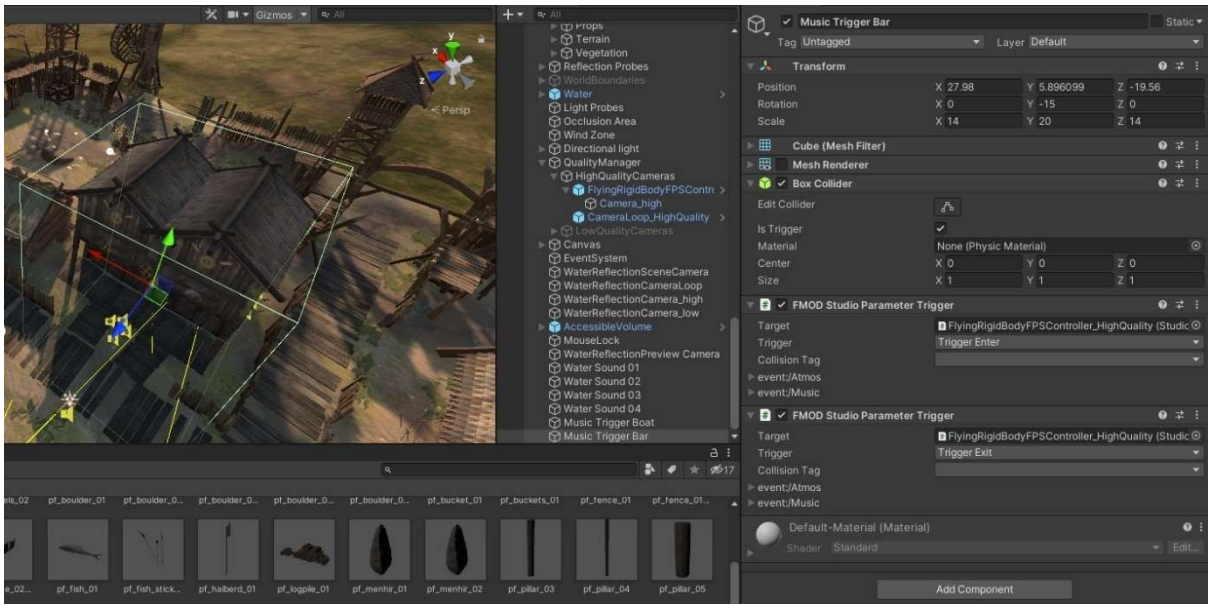

The first old song that came to mind (although not Viking) was Greensleeves. The student went on recording in a few versions of the song until he was happy with two of them, that were placed in the left and right channels but with different takes on each side. The tracks were equalized and compressed as needed.




On FMOD, the very first note of the song was left out of the loop and, to better fit the surrounding sounds such as the atmosphere, the guitar needed some extra equalization, taking out more of the low end and a bit of the extra high frequencies. The boat and the bar had their audios placed within a multi-instrument set to async. To give the impression people were *inside* the bar a low pass filter was added to represent the effect that the walls would cause in this case (cutting the high end).



In order for these changes to work on Unity, one continuous parameter ranging from 0 to 2 was added. If the parameter was set to 1 (which was its starting point), the guitar sound would be playing and all the other tracks will have their volume turn (automated) to zero. If the parameter was set to 0, the boat sound would have their volume increase as the others would turn down; and, if the parameter was set to two, the bar would have the volume turned up as the other tracks would have their volumes turned down. In Unity, this event was attached to the *FlyingRigidBodyFpsController* as an event emitter. A sphere was added around the boat and a cube was added around the bar with their mesh filters turned off and the colliders set to “triggers”. For those, two FMOD parameter triggers were added, targeting the *FlyingRigidBodyFpsController* and one being set to “trigger enter” and “trigger exit” to change the song as the player goes in and out of these areas.




In order for the player to listen to all the game sounds, in Unity, within the Camera_high, a *FMOD studio listener* was added.

# Conclusion

In conclusion this assignment was a great practical exercise where the student had the chance to develop and understand more in-depth how sounds are linked to games. Having the chance of implementing sounds in a game like this helped the student not only understand the concepts of the middleware and game engine, but also develop a better understanding on how will his sounds be implemented, changing the overall mindset when creating assets or music for games with a more matured view.

# Bibliography

Akash Thakkar (2020) Reaper for Game Audio | Part 12 - Reaper & FMOD. 9th November 2020. Available at: https://youtu.be/4xWPcS6v_6A (Acessed 30th April 2022)