Jiaye and I made a simple action game called Mike Roll Fon (microphone). The player, Mike, protects enemy cats from stealing Fon's food. It is similar to tower defense games and Vampire Survivor. The key mechanic is to use the device's microphone volume input to increase an attack range that changes colour to chase off the correspondingly coloured enemies.

The original idea for the main interactive mechanic is from another game I made in Unity3D. However, Unity's microphone input was very finicky and would work on some devices but not others. After testing out p5.js' capabilities and the microphone volume logging, I felt like it would be better as a 2D top-down action game. The characters were suggested by Jiaye and the overall presentation was a lot better this time around and is basically a new game.

Our first prototype involved drawing a circle that changed size scale under an image of a cat that could be controlled with WASD, using the cookie sketch from an earlier class activity. We made a variable that controlled the size of the cookie by logging the volume level of the microphone. As the level changes were very miniscule, we applied a multiplier to scale the value to something that could meaningfully affect the circle size. This multiplier would also eventually be controlled by the sensitivity slider to cater to different device microphones. We implemented an Enemy class and a spawning system that creates enemies offscreen and moves them towards the center of the sketch every frame. We then made it randomise colours and checked to see if the attackRing colour was the same as the enemy colour, chasing them off if their colliders overlapped. The collider logic was mostly coded in reference to the Flappy Bird project, and was a little tough to understand since in most game engines collision detection logic was done by the engine itself. On my end, I tried to minimise the usage of AI coding assistants wherever possible, using only to find bugs I couldn't recognise and work around quirks like audio, timing and css.

Originally, we wanted to use p5play to create the game, as it was basically a game engine with lots of interesting features like sprite and animation management, complex collision systems and even 3D capabilities. However, after playing around with it, we realised the most important component that we required, microphone volume interaction, was not compatible. We were also unable to get it to work alongside p5.js despite downloading both libraries. I'm sure there is a way to get both to work together since other people have done so but we figured out we were better off just working entirely in p5.js since most of the functions we needed we could just make in it.