Developer Log:

How did you mange dependencies and state in your game?

- GameState script controls state using static variables.
- BeatSpawner has a timer that determines when the player and AI are able to attack.
    - Moved some code out of BeatSpawner, but it could still be refactored further to improve Separation of Concern.
- Enemy receives player script to control pathfinding

What Systems/Patterns are used?

* Command

* A-star Pathfinding

* Object Pool

How is the game structured and why did you structure it like that?

* Input: Controls player input

* Command: Each input is given it's own command. Attacks each have their own script that contains an array of effected squares. This allows easily adding more attacks.

* Enemies: The Enemy and EnemyManager script controls all the AI, making use of the same UnitProfile/Movement/Attack scripts as the player.

* Pathfinding: Basic A* pathfinding that the enemy makes use of.

* Rhythm Scripts: BeatSpawner controls the timer and informs the Player/Enemies of when they can move. Song/Songs sort all the available songs and their bpm. UIMetronome controls the visual metronome of when the player can press.

* ObjectPool: An object pool holds all the attackMarkers. This improves game performance and allows easy sorting. This could be improved by having the EnemyManager also use an objectPool to hold the enemies.

PlayerController: Handles all the player actions and variables.

What still doesn't work, what could work better?

* BeatSpawner could be refactored further

* EnemyManager can be refactored to use ObjectPool * Enemy movement bugs could be fixed