Project Schedule

Name: Beat Breaker

Student Name: Jake Fitzgerald
Student No: C00288105

Phase 1: Core Systems & Setup (Weeks 1 – 2)

- SFML Project setup with file structure.
- Input handling:
  - Input / Event handling type (Keyboard keys, Joystick controller)
  - Latency and grace period for detecting if the input was on beat.

- Basic scene layout (Menu, Options, Gameplay, Results, etc).
- Visuals
  - Window resolution (static/dynamic)
  - Sprite sizes, consistent numbers (64 by 64 pixels, etc)
  - UI Buttons for scene navigation (clickable), Font import and basic text.

Simple character controller (physics, gravity, friction (air/grounded), collision detection)

- Movement, falling, block breaking
- Basic animations (move left/right, falling, hit)

Level generation

- Preset test level using non-randomised block patterns
- Generic block patterns used for balance
  (Plus shaped obstacle block with health in the middle [RISK/REWARD])
- Using MIDI to influence which block patterns are chosen and what blocks are used in the randomised block patterns
- (More snare hits allow for higher amounts of single non-joined blocks, less would have more joined block colours to allow for less hits needed to be used to move downwards through the level.

Timeline scrolling

- Horizonal trigger collider scrolls down through the screen vertically in time with each measure of the music track. When it reaches the end it then spawns back at the top at the start of the measure.

Input timings with MIDI track

- Test scene that prints the timings of the MIDI track for when inputs are determined to be in either the range of ‘Too soon’, ‘Perfect’, ‘Too late’ -> All considered ‘On-beat”.
- Simple HUD element component that will be used in the finished gameplay scene: Beat counting display -> Set of sf::rectangle shapes that are aligned in a row. A rectangle shape’s colour is changed from Blue to Red as the beat is aligned with said rectangle.
  Example: Common time (4/4)

]Hit the first beat]
1, 2, 3, 4
RED, Blue, Blue, Blue

[Hit the second beat]
1, 2, 3, 4
Blue, **RED**, Blue, Blue

[Hit the third beat]
1, 2, 3, 4
Blue, Blue, **RED**, Blue

[Hit the final beat]
1, 2, 3, 4
Blue, Blue, Blue, **RED**

This will also make debugging visually clearer as to when the beat is synced with the MIDI's music. Using the MIDI parser, we can also change the amount of rectangles that represent the current time signature (3/ 4, 6/ 8, etc). This makes it easier for the user to understand when they should be hitting the key press.

Phase 2: Simple prototype showcasing core gameplay (Weeks 3 - 4)
First playable demo
November Presentation

Phase 3:

Phase 4:

(After Christmas)
Phase 5: