**AI System**

A screenshot of a computer

Description automatically generated**The AIBehaviorManager**

The AI behaviors are stored in an array in the **AIBehaviorManager**. Every variable that each behavior will use is stored here as well. If any new behavior is created, it can be slotted easily in the array in the manager.

**A computer screen shot of a program

Description automatically generatedAdding New Behaviors**

The AI system allows you to create different behaviors for your AI characters. Each behavior is defined by a class that inherits from **AIBehaviorBase**, which provides three main methods: **Activate**, **Deactivate** and **UpdateBehavior**. These methods ensure smooth transitions between behaviors.

To create a new behavior, you need to:

1. Create a new class that inherits from **AIBehaviorBase**.
2. Implement the **Activate**, **Deactivate**, and **UpdateBehavior** methods in your new class.
3. Assign a unique ID to your new behavior.
4. Add an instance of the new behavior to the **AIManager** gameObject.
5. Add the new behavior class to the array of behavior in the **AIBehaviorManager**.

**A screenshot of a computer program

Description automatically generatedSetting Up The Patrol Route**

The patrol behavior is set to move the AI from point to point that is inside of a list in the **PatrolBehavior** class. To update the patrol route waypoints from the existing list, follow these steps:

Adding Waypoints:

1. To add a new waypoint to the existing list, locate the **PatrolBehavior** class in your code.
2. Find the waypoints list variable and add a new entry for the new waypoint. Ensure that the new entry is of type Transform and contains the Transform component of the new waypoint GameObject.
3. Save your changes. The AI will now patrol between the existing waypoints and the newly added waypoint.

**A computer generated image of a computer mouse

Description automatically generatedUsing Sphere Collider for Sound Detection**

A sphere collider attached to the player is used for sound detection. The collider is always active and checks if an AI is inside collider while the player presses the 'F' key. If the AI is inside the collider, the AI will check for obstacles between itself and the player. If there are no obstacles, the AI will move towards the player's position.

Below is the configuration for the sphere collider:

* Add a sphere collider component to the player’s child GameObject name **PlayerObject**.
* Configure the collider's radius to define the range of the sound detection.
* Set the sphere collider as a trigger to detect collisions without causing physical interactions.
* Position the collider appropriately to cover the area around the player where sound should be detected.