/////////////////////////////////////////////////////////////////////////////////////////////////////////

**TEMPLATE**

|  |  |
| --- | --- |
| **Use Case:** | **ID:** |
| **Description:** | |
| **Primary Actors:** | **Secondary Actors:** |
| **Preconditions:** | |
| **Main Flow:** | |
| **Postconditions:** | |
| **Alternative Flows:** | |

**TEMPLATE**

//////////////////////////////////////////////////////////////////////////////////////////////////////////

|  |  |
| --- | --- |
| **Use Case:** Enemy Detection | **ID:** 1C |
| **Description:**  An enemy will be able to detect the player slowly over time | |
| **Primary Actors:** Enemy | **Secondary Actors:** |
| **Preconditions:**   1. Player is in an undetected state | |
| **Main Flow:**   1. The use case will begin when the player enters the enemy`s field of view 2. A timer will start to check how long the player is in the enemy`s field of view 3. If the timer reaches a pre-determined endpoint, the player will be spotted 4. The timer should have a shorter pre-determined endpoint if the player is closer and more central in the enemy`s line of sight | |
| **Postconditions:**   1. The player is spotted by the enemy | |
| **Alternative Flows:** The player escapes the enemy`s field of view before the timer ends | |
| **Preconditions:**   1. The player is in the enemy`s field of view AND the timer has not ended | |
| **Alternative Flow:**   1. The player escapes the enemy`s field of view 2. The timer starts counting back down until 0 3. The player is not spotted and the timer reaches 0 | |
| **Postconditions:**   1. The enemy does not spot the player | |

|  |  |
| --- | --- |
| **Use Case:** Enemy Pathfinding | **ID:** 1B |
| **Description:**  The enemies will use a pathfinding algorithm to get to the player’s last known location. The enemies will then path find from that point outwards to try to locate the player. | |
| **Primary Actors:** Enemy | **Secondary Actors:** |
| **Preconditions:**   1. The player has been spotted by an enemy AND has since escaped | |
| **Main Flow:**   1. All of the enemies nearby, convene on the player`s last known location 2. The enemies use a pathfinding algorithm to traverse different parts and corners of the map | |
| **Postconditions:**   1. The enemies end their search after a short search and return to their pre-determined patrol paths | |
| **Alternative Flows:** N/A | |

|  |  |
| --- | --- |
| **Use Case:**  Enemy Behaviour Tree | **ID:** 1A |
| **Description:**  The enemy’s behaviour will be dictated by a behaviour tree containing 4 states. The 4 states include Patrolling, Chasing, Attacking, Searching | |
| **Primary Actors:** Enemy | **Secondary Actors:** |
| **Preconditions:**   1. There are instances of guards using the behaviour tree within the game | |
| **Main Flow:**   1. If the player has not been spotted, the guard will patrol a pre-determined route 2. When a guard spots the player, all of the guards will chase the player 3. When the guards are within range, they will attack the player 4. If the player escapes, the guards will search the area 5. If the player is found again, they will chase and attack 6. If the player is not found again, they will go back to their patrol paths | |
| **Postconditions:**   1. The states will reset back to what they were before the player was spotted | |
| **Alternative Flows:** N/A | |

|  |  |
| --- | --- |
| **Use Case:** UI Depicting Enemy States | **ID:** 2A |
| **Description:**  The enemies will have a small UI element/sprite above their heads depicting their current state. The 4 states the enemy can be in include, Patrolling, Chasing, Attacking and Searching. These will be split into 3 groups. One for Patrolling which will have no UI element. One for Searching will have a UI element/ Finally, one for Chasing and Attacking, which will have the same UI element. | |
| **Primary Actors:** Enemy | **Secondary Actors:** |
| **Preconditions:**   1. The game will be running | |
| **Main Flow:**   1. The guards will have no UI element above their heads when patrolling 2. The guards will have a small exclamation mark icon above their heads when the player has been detected and the guards are either chasing or attacking 3. The guards will have a small question mark icon above their heads when the player has been lost and they are searching | |
| **Postconditions:**   1. The guards will return to having no icons above their heads if the player has not been found and they return to patrolling | |
| **Alternative Flows:** N/A | |

|  |  |
| --- | --- |
| **Use Case:** UI Depicting Enemy Detection Level | **ID:** 2B |
| **Description:**  When the player is visible to the enemy, a small UI element will appear above the head of the enemy. This bar will fill up slowly based on close the enemy is to detect the player. The closer the enemy is to detecting the player the further the bar will fill up. The quicker the enemy is detecting the player, the faster the bar will fill up. | |
| **Primary Actors:** Enemy | **Secondary Actors:** |
| **Preconditions:**   1. The game will be running and the UI element for detection will not be above the enemy`s head. | |
| **Main Flow:**   1. The player will start in an undetected state. 2. The player will then enter the enemy’s vision zone. The detection bar should take the same amount of time to fill up as it does for the enemy to detect the player in that zone. E.g. it should take 1 second for zone 1, 5 seconds for zone 5, etc. 3. Once detected, the detection bar should disappear and reveal the alerted UI symbol from objective 2A. | |
| **Postconditions:**   1. The detection bar will only reappear when the guard is not in an alerted state and the guard is in the process of spotting the player. | |
| **Alternative Flows:**  The player goes into hiding while the enemy`s detection bar is half full. | |
| **Preconditions:**   1. The player is in an undetected state | |
| **Alternative Flow:**   1. The player will go into the enemy`s vision cone. 2. The enemy will begin to detect the player and the detection bar should start to fill up 3. Before the enemy fully detects the player, the player will leave the line of sight. 4. The detection bar should start going back down to empty if the player is not visible to the enemy. | |
| **Postconditions:**   1. The detection bar return to empty and subsequently disappears. | |

|  |  |
| --- | --- |
| **Use Case:** Smoke Bomb Mechanic | **ID:** 3ai |
| **Description:**  The player will be able to use a smoke bomb in order to evade the enemies. The smoke bomb will temporarily blind any enemies within the smoke bomb radius and all enemies will lose sight of the player while the player is within the smoke bomb radius. | |
| **Primary Actors:** Player | **Secondary Actors:** Enemies |
| **Preconditions:**   1. The player will have been spotted by the enemies and attempting to escape | |
| **Main Flow:**   1. The player will press a button on the keyboard to drop a smoke bomb at their feet 2. The smoke bomb will explode soon after it hits the ground 3. The smoke bomb will have a radius much larger than the player 4. The smoke will obstruct the vision of any enemy 5. Any enemy within the smoke will lose sight of the player and remain stationary until the smoke clears | |
| **Postconditions:**   1. After the smoke clears the enemies will go into a search state if they cannot immediately see the player | |
| **Alternative Flows:** The smoke bomb explodes while the player has not been spotted by the enemies | |
| **Preconditions:**   1. The player will be hidden from the guards | |
| **Main Flow:**   1. The player will drop a smoke bomb 2. If the guards see the smoke bomb they will investigate it 3. After the smoke clears and the guards have stopped investigating the smoke bomb, they will enter a search pattern | |
| **Postconditions:**   1. The guards will search for the player as normal | |