Hola, David. Te resumo a continuación la estructura general del programa, así como las funciones más relevantes para que puedas ubicar el contenido, y a modo de pseudo-documentación. Obviaré el contenido que has creado en clase:

class UPathFindingUtils

He implementado la función GetRandomNode() para la creación de nodos aleatorios a la hora de “spawnear” al target y los obstáculos.

class UNodeComponent:

Setea las colisiones (BlockAll / QueryOnly) en los nodos para detectar el raycast del ratón, ya que los movimientos del jugador se gestionan con mouse.

class NavGraph

EgameState:
enum que gestiona los turnos a modo de Game State.

Constructor:
Gestiona la creación de componentes (Player/Enemy/Target) y setea sus valores iniciales

BeginPlay:
Setea las posiciones iniciales de los “players , los obstáculos (finalmente lo he dejado a 0 para que aparezcan a partir del segundo round) y setea el input del jugador.

GenerateRandomObstacles()
Evalúa los nodos disponibles y excluye aquellos inválidos para la colocación de un obstáculo (Start, End y Target). Los añade a un array (BlockedNodes) y setea su material a rojo).

IsNodeWithinMoveRange()
Evalúa los nodos en función de una distancia (MaxDistance) y devuelve un booleano para saber si es accesible para el jugador o excede su capacidad de movimiento (2).

OnNodeClicked()
Verifica que el nodo sea accesible para el jugador a través de la función anterior y evalúa que no sea la posición del enemigo o un obstáculo. Si el movimiento es posible, mueve al jugador. Si en ese movimiento el nodo de destino == al del target, reubica al target y pasa al turno del enemigo.

StartEnemyTurn()
Inicia el timer a modo de delay para que el movimiento del enemigo no sea inmediato.

ProcessEnemyMove()
Utiliza las funciones Djikstra y Backtrack originales para calcular el movimiento más ventajoso entre la posición del enemigo y la posición actual del jugador. De manera innecesaria, pero para que el recálculo del path sea visible, resetea los materiales del tablero y los setea de nuevo al incio de cada turno del enemigo.

RelocateTarget()
Evalúa los nodos excluidos (player, enemy y obstaculos) y “reespawnea” el target en un nodo aleatorio. Aquí he tenido que volver a ejecutar Djikstra y backtrack porque ha sido la única forma que he encontrado de volver a resetear los materiales del grafo y volver a representar los obstáculos y el path correctamente. De lo contrario tenía un baile de obstáculos que no sabía cómo resolver (spoiler: esa es también la razón por la que los obstáculos en beginPlay están seteados a 0, porque estaba buscando el origen del problema. Finalmente me ha parecido que tenía sentido que en el primer nivel no hubiera obstáculos y que aparecieran a partir del segundo y lo he dejado así 🙂)