First Deliverable: Game Development Project

Due: Friday, May 9th, 12:00 noon

### **Game Design Document (GDD)**

#### **Game Concept & Overview:**

**Project Title:** AI Pathfinding Showcase

**Elevator Pitch:** Interactive 10x10 grid BFS visualizer to teach fundamentals of graph search

**Objectives:** AI clarity, step by step animation

#### **Scope of Vertical Slice:**

* 10x10 Grid centred at screen
* Place Start (green) and End (magenta) tiles
* Toggle walls (red) by clicking remaining tiles
* Four UI controls (Select Start, Select End, Reset Grid, Run BFS)
* Visual BFS wave and path, plus moving agent

#### **Core Mechanics:**

**Tile Interaction:** Click behaviour changes based on mode flags (selectingStart, selectingEnd)

**Obstacle Editing:** Toggle walkable state on standard clicks

**Search Execution:** BFS executed by pressing “Run BFS,” visualized via colour waves

#### **User Interface (UI/UX):**

**Controls Panel:** Vertical panel in top-left with rounded‐corner buttons

**Colour Palette:** High contrast (white grid, dark background, coloured states)

**Feedback:** Immediate tile colour changes, delay controls for pacing

#### **Visual Style:**

Flat design with no textures

Colour‐coded states:

* white (walkable),
* red (blocked),
* green/magenta (start/end),
* orange/yellow (wave),
* green/blue (path)

#### **Player Experience Goals:**

* Clear visual learning of BFS order-of-visit
* Engaging, with control over demonstration speed
* Encourages experimentation with obstacle layout

### **Technical Design Document (TDD)**

#### **Development Stack:**

* Unity 2021.3 LTS
* C# MonoBehaviour scripts
* TextMeshPro for button labels

#### **Architecture Overview:**

**GridManager:** Generates grid, handles clicks, BFS coroutine, agent movement

**Tile:** Encapsulates state flags, colour updates, click delegation

**UI:** Four Button components wired to GridManager methods via onClick listeners

#### **Grid & Tile Implementation**

* Tile[,] tiles array created in GenerateGrid() with world‐space offsets
* Tiles instantiated from Tile prefab containing a SpriteRenderer

#### **Input Handling & Selection Modes**

* Modes toggled by SelectStartMode() and SelectEndMode()
* HandleTileClick(Tile t) inspects mode flags and either sets start/end or toggles wall

#### **BFS Algorithm & Timing**

Coroutine RunBFS\_Visual() implements BFS using:

* Queue<Vector2Int> for frontier
* bool[,] visited and Vector2Int[,] parent arrays
* Adjustable delays: waveStepDelay, pathStepDelay (serialized fields)
* Early exit when end is dequeued

#### **Agent Movement**

MoveAgentAlong(List<Vector2Int> path) lerps agentInstance between tile positions, using Time.deltaTime over fixed durations

#### **UI Integration**

Buttons referenced via public Button fields

Added listeners in Start():

* Btn\_SelectStart.onClick.AddListener(SelectStartMode);
* Btn\_SelectEnd.onClick.AddListener(SelectEndMode);
* Btn\_ResetGrid.onClick.AddListener(ResetGrid);
* Btn\_RunBFS.onClick.AddListener(RunBFSVisualWrapper);

Buttons disabled/re-enabled around BFS execution.

### **Future Enhancements**

* Add A\* option with heuristic parameter
* Allow grid resizing at runtime
* Performance: object pooling for agent and tiles
* Improved UI polish, responsive layouts