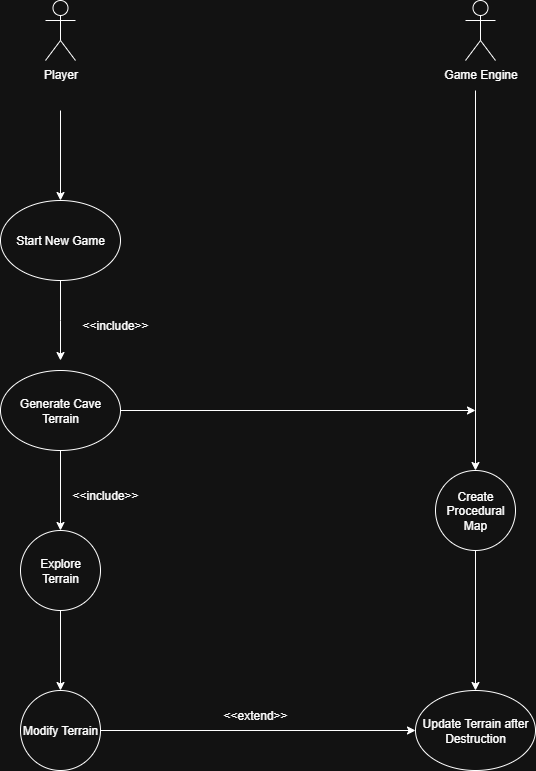
Name **Matthew Brisset** Mark \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/50

## Brief introduction \_\_/3

My feature is procedurally generated terrain with realtime destruction. It will allow the player to generate a world based on a seed value. It will procedurally generate chunks in a grid, with each chunk having cave-like geometry. The reason for using a chunk system is performance reasons. This feature will also have the ability for the player to modify the terrain in real-time using tools by calling functions upon the world handler. Each time the world is modified, it will update and re render the chunks involved.

## Use case diagram with scenario \_\_14

### Use Case Diagrams



### Scenarios

**Name:** Start New Game

**Summary:** The player selects ‘new game’ from the menu.

**Actors:** Player, game engine

**Preconditions:** Game is installed, player is at menu

**Basic sequence:**

**Step 1:** Select new game from menu

**Step 2:** New game session is created.

**Step 3:** Game engine generates cave terrain

**Step 4:** Cave terrain is displayed to player

**Exceptions:**

**Step 1:** Terrain generation fails -> Error displayed, tries again

**Post conditions:** Player is placed in new cave

**Priority:** 1

**ID:** C01

\*The priorities are 1 = must have, 2 = essential, 3 = nice to have.

**Name:** Explore terrain

**Summary:** The player navigates through the cave

**Actors:** Player

**Preconditions:** A game session is running with the generated cave

**Basic sequence:**

**Step 1:** Player moves character through the cave

**Step 2:** System updates player position

**Step 3:** Render new environment

**Step 4:** Player can encounter walls preventing movement

**Exceptions:**

**Step 1:** Player collides with invalid terrain -> system corrects position

**Post conditions:** Players position in cave is updated

**Priority:** 2

**ID:** C02

\*The priorities are 1 = must have, 2 = essential, 3 = nice to have.

**Name:** Destroy terrain

**Summary:** The player destroys a section of cave using tools

**Actors:** Player, game engine

**Preconditions:** Terrain is present and marked as destructible

**Basic sequence:**

**Step 1:** Player uses tool to destroy terrain

**Step 2:** System detects terrain destruction

**Step 3:** Section of terrain is removed

**Step 4:** Terrain generator recalculates geometry and collision bounds

**Step 5:** Render updated terrain

**Exceptions:**

**Step 1:** Terrain is indestructible or out of bounds -> ignore destruction

**Step 2:** Terrain fails to update -> Revert to previous terrain state

**Post conditions:** Terrain is modified and re-rendered

**Priority:** 2

**ID:** C03

\*The priorities are 1 = must have, 2 = essential, 3 = nice to have.

**Name:** Update terrain after destruction

**Summary:** The game engine updates terrain geometry and collisions when destruction occurs

**Actors:** Game engine

**Preconditions:** A section of terrain is destroyed

**Basic sequence:**

**Step 1:** Game engine detects destruction

**Step 2:** Recalculate mesh and collision bounds

**Step 3:** Render new terrain on player view

**Exceptions:**

**Step 1:** Terrain update fails or is out of bounds -> Attempt to revert

**Post conditions:** Cave mesh and collisions are updated to reflect destruction

**Priority:** 2

**ID:** C04

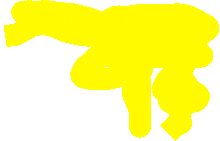
\*The priorities are 1 = must have, 2 = essential, 3 = nice to have.

## Data Flow diagram(s) from Level 0 to process description for your feature \_\_\_\_\_\_\_14

Data Flow Diagrams

A diagram of a game

AI-generated content may be incorrect.



### Process Descriptions

**Generate Cave Terrain (On new game started):**

SET randomSeed = seed

CREATE grid[width][height]

FOR x = 0 TO width:

FOR y = 0 TO height:

value = RANDOM(randomSeed)

IF value < threshold THEN

grid[x][y] = "Solid"

ELSE

grid[x][y] = "Empty"

**Render Single Chunk:**

FOR x = chunkX TO chunkX + chunkWidth:

FOR y = chunkY TO chunkY + chunkHeight:

blockType = grid[x][y]

IF blockType == "Solid" THEN

DRAW "Stone"

ELSE IF blockType == "Empty" THEN

DRAW "Air"

**Manage Multiple Chunks:**  
FOR each chunk IN allChunks:

distance = CALCULATE\_DISTANCE(chunk.center, player.position)

IF distance <= renderDistance THEN

chunk.status = "Loaded"

ELSE

chunk.status = "Unloaded"

**Apply Textures:**  
FOR each block IN chunk:

IF block == "Solid" THEN

block.texture = "StoneTexture"

ELSE IF block == "Empty" THEN

block.texture = "AirTexture"

**Destroy terrain:**  
targetBlock = grid[actionX][actionY]

IF targetBlock == "Solid" THEN

grid[actionX][actionY] = "Empty"

**Update Terrain after Destruction:**

CLEAR chunk mesh

CALL RenderChunk for that chunk

## Acceptance Tests \_\_\_\_\_\_\_\_9

[Describe the inputs and outputs of the tests you will run. Ensure you cover all the boundary cases.]

|  |  |  |
| --- | --- | --- |
| **Case** | **Setup** | **Expected output** |
| **Chunk generation** | **Generate a chunk** | **Nodes are generated for the chunk. Ensure that no out of bounds nodes are accessed.** |
| **Chunk rendering** | **Render a generated chunk** | **Geometry is created based on nodes. Ensure that triangles, vertices, etc are correct.** |
| **Chunk destruction** | **Destroy nodes in a chunk** | **Nodes are set to FALSE, geometry is re-rendered. Ensure that no out-of-bounds nodes are accessed.** |

## Timeline \_\_\_\_\_\_\_\_\_/10

[Figure out the tasks required to complete your feature]

Example:

### Work items

|  |  |  |
| --- | --- | --- |
| Task | Duration (hrs) | Predecessor Task(s) |
| 1. Single cave chunk renderer | 3 |  |
| 2. Randomize cave terrain | 1 | 1 |
| 3. Terrain destruction | 4 | 1, 2 |
| 4. Chunk textures | 2 | 1 |
| 5. Multi-chunk handler | 3 | 1 |

### Pert diagram

A diagram of a path

AI-generated content may be incorrect.

### Gantt timeline

