Objective:
To create a system in Unity that generates terrain procedurally (using Perlin noise, Voronoi, or other algorithms), supports biome-based environments, and integrates basic gameplay mechanics (exploration, resource collection, or survival) demonstrating practical use of terrain.

Mentor-Guided Deliverables

Phase 1: Foundation & Research

Deliverable 1.1: Research Report

- Types of procedural terrain:
    - Heightmap-based (Perlin noise, fractal noise)
    - Mesh-based
    - Voxel-based (optional for advanced teams)
- Terrain features: Hills, valleys, rivers, caves.
- Tools to use: Unity Terrain API vs Mesh generation, Noise libraries (LibNoise, FastNoise).
- Examples from games (e.g., Minecraft, No Man's Sky, Valheim).

Deliverable 1.2: Base Project Setup

- Empty Unity scene with camera and movement system.
- Choose rendering pipeline (URP/HDRP).
- Select approach: Unity Terrain tool or Mesh generation.

Phase 2: Terrain Generation Algorithms

Deliverable 2.1: Initial Terrain Generation

- Use **Perlin noise** to generate a heightmap.
- Allow adjustable parameters: scale, amplitude, frequency.
- Create a TerrainSettings scriptable object or UI menu to tweak generation in real time.

Deliverable 2.2: Multi-layered Generation

- Add support for:
    - Multiple noise layers for realistic detail.
    - Biome detection (e.g., snow above Y = 30, sand below Y = 10).
    - Texture painting based on height or slope.

Phase 3: Visual & Performance Enhancements

Deliverable 3.1: Terrain Optimization

- Add **chunk-based terrain loading** (LOD).
- Implement **terrain culling** or pooling system for infinite scrolling terrains.
- Optional: Add support for multithreading using Unity's Jobs or Coroutines.

Deliverable 3.2: Terrain Shader & Materials

- Apply materials using Shader Graph.
- Blend multiple terrain textures (grass, rock, snow) using slope and height maps.
- Add ambient elements like fog, skybox, dynamic lighting.

Phase 4: Gameplay Integration

Mini Gameplay Concept:
A first-person exploration and resource-collection game where players navigate and survive on procedurally generated terrain.

Deliverable 4.1: Character Controller

- Add FPS or 3rd-person character.
- Allow jumping, sprinting, and climbing slopes.

Deliverable 4.2: Gameplay Mechanics

- Spawn interactable objects based on terrain (e.g., trees on flat ground, stones on slopes).
- Collectibles or crafting system (e.g., collect wood and craft a torch).
- UI system to display player inventory and stats.

Deliverable 4.3: Terrain Interaction

- Ability to dig, modify, or destroy parts of the terrain.
- Drop objects that stick to or float on terrain.

Phase 5: Biomes & Random Events (Advanced)

Deliverable 5.1: Procedural Biomes

- Different terrain types: Desert, forest, snowfields, mountains.
- Use Perlin noise for biome mapping.
- Each biome has unique textures, resources, and music.

Deliverable 5.2: Events or Challenges

- Add survival mechanics: day-night cycle, weather (rain affects movement).
- Spawn enemies or hazards depending on biome.
- Add checkpoints or objectives tied to terrain features (e.g., find cave on highest mountain).

Phase 6: UI, Menu & Replayability

Deliverable 6.1: Main Menu

- Options to seed terrain, select difficulty (affects terrain ruggedness or hazard spawn rate).

Deliverable 6.2: Saving & Regeneration

- Option to **regenerate terrain using the same seed**.
- Save system for terrain state and player progress.

Evaluation Rubric

Criteria	Description	Weight
Terrain Generation Quality	Realism, noise layering, biome variation	25%
Visual Quality & Optimization	Use of shaders, LOD, performance handling	20%
Gameplay Mechanics	Integration with terrain (exploration/resource)	20%
Customization & UI	Seed input, parameter sliders, menu	15%
Creativity & Innovation	Unique terrain features or mechanics	10%
Documentation & Demo	Code clarity, final walkthrough	10%