Project Proposal

Operation Clean Sweep – Tactical Shooter Game (TPS/FPS Hybrid)

Submitted By:

Muhammad Affan bin Aamir – Project Lead, Gameplay & Systems Developer

Umar – Environment & Assets Coordinator, Blueprint Support

Course: Data Structures & Algorithms (DSA) Project

Semester: Summer 2025

Date of Submission: July 3, 2025

Contents

1 Overview

Operation Clean Sweep is a tactical third-person shooter game (with a toggle to first-person mode), built using Unreal Engine 5. The game places the player in command of a 4-member squad (1 player + 3 AI bots, with multiplayer as a stretch goal) tasked with assaulting an enemy base defended by 25 AI enemies, followed by 15 reinforcement bots. It aims to provide a rich blend of gameplay and algorithmic depth by embedding Data Structures and Algorithms concepts directly into its systems.

This game is both a technical and creative challenge, helping the team understand game development fundamentals while applying computer science theory.

2 Objective

To build a playable shooter experience that demonstrates:

- Team-based AI combat with 25 initial enemies and 15 reinforcements
- Smooth player mechanics with TPS/FPS toggle
- Real-time stat tracking and UI (mini-map, Tab menu)
- Direct implementation of Data Structures like Arrays, Queues, Stacks, Linked Lists, Trees, Graphs, Hash Maps, State Machines

3 Key Features

- TPS to FPS toggle (camera + combat mode switch)
- Combat mechanics (shooting with hip fire spread, scoped FPS view, limited healing)
- AI-controlled enemies with patrol, alert, attack, and reinforcement behaviors
- Teammates (1 player + 3 bots) with individual stat tracking
- Custom UI for health, ammo, team stats (Tab View), and mini-map (expandable via 'M')
- Dynamic environment with death zones (river), HQ assault zone, ammo/healing pickups

4 Tools & Technology

- **Game Engine:** Unreal Engine 5 (C++ + Blueprints)

- **Languages:** C++, Blueprint Scripting
- **Assets:** Unreal Marketplace, Mixamo, Quixel Bridge (for map, characters, weapons)
- **Version Control:** GitHub
- **Documentation & Planning:** Notion

5 Learning Goals

- Master basic to intermediate game development in Unreal Engine
- Learn gameplay programming and animation handling in C++/Blueprints
- Apply DSA concepts practically in game mechanics
- Improve project planning and team coordination using industry tools

6 DSA Concepts to be Implemented

Concept	Use Case
Arrays	Storing squad members, enemy pools, bullet tracking
Queues	Managing spawn timers, healing cooldowns, reinforcement wave scheduling
Stacks	Handling undo actions for player inputs, recent damage event tracking
Linked Lists	Maintaining dynamic lists of active AI bots or ammunition pickups
Trees	Implementing Behavior Trees for enemy AI logic (patrol, alert, attack)
Graphs	Utilizing Navigation Mesh for AI pathfinding
Hash Maps	Tracking kills, points, assists, and damage stats per player
State Machines	Managing player states (idle, shooting, sprinting, prone) and game states (start, playing, win/loss)

7 Development Timeline

Sprint	Dates	Focus
Sprint 1	Jul 5–11	UE5 Setup, Character Controls
Sprint 2	Jul 12–18	Player Combat & Scope
Sprint 3	Jul 19–25	Map Setup & AI Behavior
Sprint 4	Jul 26–Aug 1	Game Systems & DSA Integration
Sprint 5	Aug 2–8	Testing, Polish, Documentation

8 Team Roles

- **Affan:** Lead Developer, Game Logic, Combat, DSA Implementation
- **Umar:** Asset Research & Integration, Blueprint Setup, Visual Polish

9 Expected Deliverables

- Playable Level 1 Demo with 4-player squad & enemy HQ
- Source code with DSA-implemented gameplay logic
- Game Design Document (GDD)
- Technical Documentation (DSA code explanation)
- Sprint Logs & Weekly Progress
- GitHub Repository

10 Success Criteria

- All core mechanics functional and tested
- Minimum one playable level with enemy AI and reinforcements
- Clear implementation of at least 4 DSA concepts in gameplay
- Clean and professional project documentation
- Completion of team coordination via GitHub and Notion