Tanks!

Project Report

Supervisor

Mr. Hafiz Muhammad Usman

Submitted by

AbdulRehman Zahid
Student id: AF23LHB3047

Afzaal Ahmad
Student id: AF23LHB3292

Riphah International College, <u>Harbanspura Campus</u>

ADP Program	Computer Science
Project ID *(for office use)*	
Title of Project	Tanks!
Type of project	[✓] Traditional [ ] Industrial [ ] Continuing
Nature of project	[✓] **D**evelopment [ ] **R**esearch [ ] **S**urvey
Area of specialization/ Field	Combat Games

Project Group Members

Sr.#	Reg. #	Student Name	CGPA	Email ID	Phone #	Signature
(i)	TeamLead AF23LHB3047	AbdulRehman Zahid	3.70	mrar08523@gmail.com	+92 342 4057037	[illegible]
(ii)	AF23LHB3292	Afzaal Ahmad	3.53	afzaal78002.pjb@gmail.com	+92 313 1499910	[illegible]

Declaration: Project group members have cleared all prerequisites courses For project as per their degree requirements.

Supervisor Name and Signatures: **Principal:**

Hafiz Muhammad Usman PROF. IZHAR AKRAM

ABSTRACT

The project **Tanks!** is a 3D tactical combat game developed using the Unity engine, designed to deliver an immersive and skill-based gaming experience. Players take control of a tank and engage in strategic battles against AI-controlled enemies within confined, destructible arenas. The game draws inspiration from classic tank battle titles but incorporates modern mechanics, enhanced visuals, and progressive level-based challenges. The primary objective is to demonstrate core game development principles while creating an entertaining and polished product.

Key features of **Tanks!** include realistic tank movement and turret rotation mechanics, physics-based projectile firing, and dynamic collision systems. The game environment consists of multiple arena-style levels, each with unique layouts, obstacles, and increasing difficulty to maintain player engagement. A scoring system, health management, and win/lose conditions ensure structured gameplay progression. Optional power-ups, such as temporary shields and speed boosts, add strategic depth. The user interface is designed for clarity, featuring a heads-up display (HUD) for health, score, and level tracking, along with menu and pause functionalities.

To enhance immersion, **Tanks!** integrates audio effects for movement, shooting, and explosions, alongside visual animations for combat feedback. Performance optimization techniques ensure smooth gameplay across different hardware specifications. The project also explores Unity’s capabilities in physics simulation, AI behavior scripting, and 3D asset integration.

Beyond its entertainment value, **Tanks!** serves as a foundational exercise in game development, covering essential aspects such as level design, player controls, and system optimization. Future expansions could include multiplayer modes, additional tank customization, and more complex AI behaviors. By combining technical execution with engaging gameplay, this project aims to provide both a learning platform for developers and a fun, challenging experience for players.

Table of Contents

1. Introduction

Tanks! is a tactical 3D tank combat game developed using Unity, where the player controls a tank and battles against enemy tanks in a confined arena. The gameplay is inspired by the classic tank battle games but enhanced with modern visuals, improved mechanics, and level-based challenges. The player must strategically move, aim, and shoot to survive and eliminate all opponents within the battlefield.

The game features a single-player campaign mode as well as potential for local or online multiplayer battles in future versions. Coupled with dynamic lighting, particle effects, and immersive sound design, **Tanks!** delivers a visually engaging and adrenaline-fueled tank combat experience that blends retro inspiration with modern game design.

2. Objective

The objective of this project is to design and develop a fun and skill-based tank combat game that emphasizes player reflexes, control, and environmental awareness. This project will also demonstrate core game development concepts in Unity, including physics-based shooting, collision handling, level design, and user interface implementation.

3. Problem Description

Tanks! is a level-based 3D tank combat game developed in Unity, inspired by classic arcade tank games but reimagined with modern visuals, mechanics, and strategic gameplay. The objective is to provide an engaging and educational game development experience while delivering a fun, tactical challenge for players.

What makes this project relevant is the current gap between overly simple retro tank games and complex, high-end titles. Most existing games either lack immersive graphics or meaningful progression, which leads to repetitive gameplay. **Tanks!** bridges this gap by introducing strategic movement, destructible environments, power-ups, and enemy AI that increases in difficulty across levels.

The **why** lies in both entertainment and education. From a gaming perspective, players seek challenging games that require planning and quick reflexes. From a development perspective, this project offers a hands-on opportunity to learn Unity game development, including physics, AI behavior, level design, and user interaction.

Designing such a system involves careful consideration of gameplay balance, performance optimization, and fault tolerance. For instance, buggy collisions or unpredictable AI could frustrate users. Smooth controls, responsive UI, and scalable architecture are key to player retention and future expansion, such as multiplayer modes or mobile ports.

In summary, **Tanks!** addresses the need for a well-balanced, modern tactical game that also acts as a strong foundation for learning and extending 3D game development practices.

4. Methodology

To tackle the problems identified in the game design, **Unity Engine** will be used as the primary development platform due to its robust support for 3D environments, physics simulation, and cross-platform deployment. Scripting will be done in **C#**, enabling full control over tank behavior, enemy AI, shooting mechanics, and game logic.

For movement and physics interactions, Unity's **Rigidbody** and **Collider** components will be used, ensuring realistic tank dynamics, collision handling, and projectile impacts. Enemy behavior will be implemented using simple **Finite State Machines (FSM)** to simulate patrolling, chasing, and attacking actions. As the levels progress, AI complexity will increase to create a more challenging experience.

For visual enhancements, **Particle Systems** and **Post-Processing Stack** will be applied to provide effects such as explosions, smoke, lighting, and camera effects. Level progression will be handled using Unity's **Scene Management** system.

To ensure performance and reduce bugs, the game will follow a modular structure using **prefabs**, **scriptable objects**, and **event-driven programming**. UI elements like health bars, ammo counters, and win/loss screens will be built using Unity's **Canvas System**.

Optional features like sound effects and background music will use **Unity's Audio Mixer**, and assets will be either self-created or sourced from free Unity Asset Store packages, following copyright guidelines.

This structured approach ensures that the game is scalable, maintainable, and delivers an engaging tactical experience.

5. Project Scope

The scope of the **Tanks!** project is limited to the development of a **single-player, level-based** 3D tank combat game with basic AI, tactical gameplay, and visual effects. The game will focus on strategic movement, enemy elimination, and level progression within closed arenas.

Multiplayer functionality, online leaderboards, and cross-platform deployment (such as mobile or web versions) are **out of scope** for the current project phase. Similarly, advanced AI techniques such as machine learning or complex pathfinding algorithms (e.g., A*) will not be implemented. Instead, simple rule-based enemy behavior will be used to maintain focus and manage project complexity.

Customization features like tank skins, in-game currency, or user profiles are also excluded. We assume that the game will be run on a **local desktop environment**, with hardware capable of handling 3D graphics at a basic level.

Our main assumption is that the project will evolve under **resource and time constraints typical of a student-level development cycle**, which limits the inclusion of large-scale assets or multiplayer infrastructure. The scope is intentionally kept focused to ensure a **functional, playable, and polished prototype** that demonstrates key gameplay mechanics, level design, and Unity development practices.

6. Feasibility Study

After careful consideration of the project's scope and objectives, the development of **Tanks!** is considered highly feasible within the timeframe and resource limitations of a Final Year Project. The game is designed to be modular, scalable, and achievable within a student-level development environment. Below is a detailed breakdown of potential risks and resource needs.

---

i. Risks Involved:

1. **Time Management Risk:**
    - Balancing project tasks with other academic responsibilities can cause delays.
    - Unexpected personal commitments, exam schedules, or coursework overlaps may affect progress.
    - **Mitigation:** A detailed Gantt chart will be created with weekly development goals and buffer time for revisions. Early prototyping and weekly check-ins will help stay on track.
2. **Technical Complexity:**
    - Implementing enemy AI, dynamic level design, and physics-based mechanics may become challenging.
    - There could be integration issues between different modules (e.g., UI, gameplay logic, level transitions).
    - **Mitigation:** Begin with minimal viable functionality and build iteratively. Use Unity's built-in physics, AI scripting patterns like Finite State Machines, and test each feature independently before merging.
3. **Team Collaboration Risks:**
    - Miscommunication, uneven workload distribution, or versioning conflicts could arise when working in a team.
    - **Mitigation:** Tasks will be clearly divided based on individual strengths. Communication will be maintained via regular in-person or online meetings, and all code/assets will be shared using **GitHub** to track changes and resolve conflicts.
4. **Scope Creep:**

- There may be a temptation to add extra features beyond the scope (like multiplayer or advanced AI), leading to time/resource strain.
- **Mitigation:** Scope will be strictly adhered to, and any new features will be logged for future implementation but not included in the initial version.

---

ii. Resource Requirements:

1. **Computing Resources:**
    - A desktop or laptop with:
        - Minimum **8 GB RAM**
        - Dedicated **graphics card (GPU)** capable of supporting Unity's 3D rendering
        - At least **50 GB free disk space**
        - Stable **internet connection**
    - These requirements are within reach for most academic environments.
2. **Software Tools:**
    - **Unity Engine (LTS version)** – for 3D game development, physics, rendering, and UI.
    - **Visual Studio** – for C# scripting and debugging.
    - **Git & GitHub** – for version control and team collaboration.
3. **Digital Assets:**
    - Free or open-source:
        - 3D tank models, terrain textures, and props from **Unity Asset Store**.
        - Audio files like background music, shooting sounds, and explosion effects from **Freesound.org**.
    - All assets will be used under appropriate licenses to avoid copyright issues.
4. **Human Resources:**
    - A dedicated team including:
        - **Team Lead/Developer** – Handles gameplay mechanics, Unity integration.

- **Tester (shared role)** – Ensures functionality through ongoing playtesting and bug reporting.

5. **Learning Resources:**
    - Online tutorials, Unity documentation, YouTube, and courses (e.g., Unity Learn, Coursera) to help solve technical issues and guide development.

7. Solution Application Areas

The **"Tanks!"** game project, while primarily designed as an entertainment product, has real-world value across multiple application domains, particularly in the fields of **gaming, education, simulation, and interactive training**.

- *Gaming Industry*

The primary application of this project lies in the **commercial and indie gaming industry**, where there is ongoing demand for engaging, skill-based tactical games. "Tanks!" delivers a blend of nostalgic gameplay inspired by classic tank battles with modern mechanics, offering potential for monetization on platforms such as Steam or mobile app stores. The modularity of the game (levels, power-ups, AI, etc.) also supports future expansion into multiplayer or co-op modes, further extending its market potential.

- *Game Development Education*

This project serves as a **valuable learning resource** for students and beginner developers interested in Unity and 3D game development. It demonstrates key concepts like physics-based shooting, collision handling, optimization techniques, and user interface design. The game can be adapted as a teaching tool or workshop material in academic institutions offering game development courses.

- *Simulation & Tactical Training*

Though simplified for entertainment, the mechanics of spatial awareness, strategic movement, and reaction timing can be repurposed in **military training simulations** or **robotics pathfinding research**. By increasing realism and integrating sensors or telemetry data, a future version of the game could contribute to controlled simulation environments for training or prototyping.

Overall, the project blends creativity and technical skill in a way that benefits not just entertainment consumers, but also learners and simulation developers across related industries.

8. Requirements:

The following outlines the **basic hardware and software requirements** necessary to design, develop, and run the **"Tanks!"** 3D tactical game project. These initial specifications ensure smooth development and gameplay experience. A more detailed breakdown will be provided later in the Software Requirements Specification (SRS) document.

Hardware Requirements

- **Development Machine:**
  - **Processor:** Intel Core i5 or AMD Ryzen 5 (or higher)
  - **RAM:** Minimum 8 GB (16 GB recommended for smoother development with Unity)
  - **Graphics Card:** NVIDIA GTX 1050 / AMD Radeon RX 560 (or higher)
  - **Storage:** At least 20 GB free space (for Unity, assets, and project files)
  - **Display:** Full HD (1920x1080) resolution monitor
- **Testing Machine:**
  - Mid-range PC or laptop with at least 4 GB RAM, integrated GPU (e.g., Intel UHD), and dual-core processor to ensure performance testing on lower-end systems.

Software Requirements

- **Game Engine:** Unity (latest stable LTS version)
- **IDE/Editor:** Visual Studio (with Unity tools) or JetBrains Rider (optional)
- **Graphics Tools:** Blender (for any custom 3D models), GIMP or Photoshop (for textures/UI assets)
- **Version Control:** Git (with GitHub or GitLab for repository management)
- **Audio Tools:** Audacity or other free tools for sound editing (if required)
- **Operating System:** Windows 10 or above (development and deployment)

These basic resources are sufficient for prototyping, building, testing, and presenting the final game. The requirements may be updated as features are added during development.

8.1. Stakeholders:

In the development of **“Tanks!”**, several stakeholders play an important role in shaping and evaluating the project.

- **End Users**: Casual gamers and strategy lovers who will play the game. Their experience determines the game’s success.
- **Project Team**: Developers and designers responsible for creating game mechanics, visuals, and overall functionality.
- **Academic Supervisors**: Guide the project, ensure it meets educational goals, and evaluate the final outcome.
- **Evaluators/Jury**: Review the final product for innovation, usability, and technical quality.
- **Future Developers**: May use this project for learning or further development, requiring clean code and documentation.

9. Tools/Technology

The development of **“Tanks!”** requires the following hardware and software tools and technologies:

Software Tools:

- **Unity Engine** – Primary game development platform used for designing gameplay, physics, animations, and level design.
- **C# Programming Language** – Used for scripting all gameplay mechanics and logic within Unity.
- **Visual Studio** – Integrated development environment (IDE) for writing and debugging C# code.
- **Git / GitHub** – For version control and project collaboration.

Hardware Requirements:

- **Personal Computer (Windows/macOS)** – For development and testing.
- **Graphics Card (2GB or higher)** – To support smooth rendering and gameplay in Unity.
- **Input Devices** – Mouse and keyboard for control testing and gameplay interaction.

10. Expertise of the Team Members

The development of **“Tanks!”** is driven by a dedicated team of two members, each bringing unique strengths and complementary skills essential for the project’s success. Below is a breakdown of their core areas of expertise:

i. Game Development and Visual Design

AbdulRehman Zahid (Team Leader)
AbdulRehman has over 18 months of experience as a graphic designer, with a strong command of visual aesthetics, creative direction, and multimedia content. He is responsible for the development of the game in Unity and manages the overall design aspects including:

- UI/UX design for immersive user interaction
- Creation and implementation of 3D models and animations
- Integration of sound effects and background music
- Ensuring visual consistency and quality across levels
  His ability to multitask ensures efficient progress in design, development, and gameplay tuning.

ii. Quality Assurance and Documentation

Afzaal Ahmad (Team Member)
Afzaal plays a key role as a game moderator and tester. With experience in moderating various Chinese games, he brings deep insight into balancing gameplay and identifying usability issues. His contributions include:

- Back-end testing of game functionality and levels
- Bug detection and gameplay refinement
- Thorough documentation of the game mechanics, progress, and testing results
  He also ensures the delivery of comprehensive and well-structured documentation, aligning with academic and technical requirements.

This combination of development, visual design, testing, and documentation expertise equips the team to complete the project effectively while maintaining high standards of quality and gameplay experience.

11. Timeframe

The development of **“Tanks!”** is planned over a total duration of **10 weeks**, ensuring focused progress and timely delivery in line with academic deadlines.

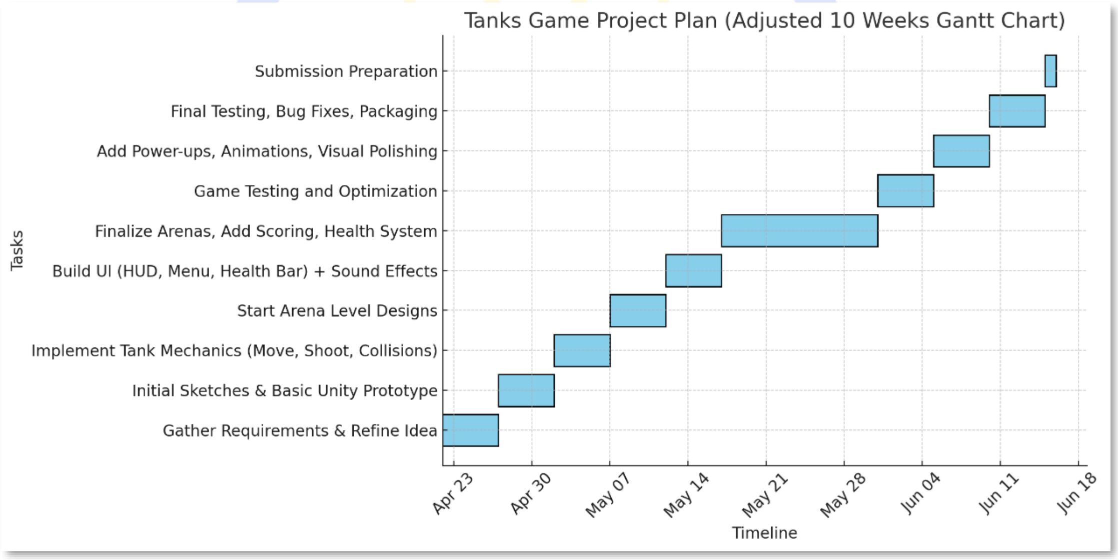
The project will follow the timeline below:

- **Week 1–2:** Requirement gathering, idea refinement, initial sketches, and basic prototyping in Unity.
- **Week 3–4:** Implementation of core tank mechanics including movement, shooting, and collisions. Start working on level designs.
- **Week 5:** Integration of user interface (HUD, main menu, health bar) and sound effects for shooting, movement, and environment.

- **Week 6–7:** Designing and finalizing arena-style levels, adding scoring system, player health, and win/loss conditions.
- **Week 8:** Testing gameplay loop, identifying bugs, and optimizing for smooth performance.
- **Week 9:** Final touches including power-ups, animations, visual polishing, and ensuring a complete game flow.
- **Week 10:** Final testing, documentation, packaging the game build, and preparing for submission.

Completing the project within this 10-week period is important because:

- **Academic Evaluation:** Aligns with university FYP schedule and ensures on-time presentation and grading.
- **Focus & Efficiency:** Shorter deadlines promote efficient task division and goal-oriented development.
- **Relevance:** Staying within timeframe helps the game remain consistent with its initial scope, ensuring quality without overextension.



12. Risks & Issues

Risks

a. Delay in implementing core tank mechanics like movement and shooting.

b. Bugs in collision detection could impact gameplay quality.

c. Incomplete level designs due to time constraints.

d. Hardware limitations may cause performance issues on low-end devices.

Issues

a. Learning curve in Unity development slowing early progress.

b. Difficulty in integrating sound effects smoothly.

c. Coordination challenges between team members on assets and testing.

Project Risks		
Risk	**Details**	**Likelihood**
Delay in core mechanics	Tank movement and shooting might take longer to finalize.	Medium
Bugs in collision detection	Faulty collisions can ruin gameplay experience.	High
Incomplete level designs	Not all three arenas (Jungle, Desert, Moon) might be finished.	Medium

Project Issues		
Issue	**Details**	**Impact**
Unity learning curve	Initial slow progress as team sharpens Unity skills.	Moderate
Sound integration difficulties	Trouble syncing sounds with player actions and events.	Moderate
Team coordination challenges	Delays in asset delivery and testing feedback loops.	High

13. Milestones

1. **Requirements Gathering & Planning**
   - Finalize game concept, core features, and arena ideas.
2. **Core Mechanics Implementation**
   - Develop tank movement, shooting, and collision handling.
3. **Basic Arena Design**
   - Create initial designs for Jungle, Desert, and Moon arenas.
4. **User Interface and Sound Integration**
   - Add HUD elements like health bar and ammo counter, and integrate sound effects.
5. **Level Finalization**
   - Complete arena designs with obstacles and environmental enhancements.
6. **Game Testing and Optimization**

- Conduct bug fixing, gameplay balancing, and performance improvements.

7. **Final Features Implementation**
    - Implement scoring system, power-ups, and win/loss conditions.
8. **Final Testing, Packaging, and Documentation**
    - Final round of testing, documentation writing, and preparing the final game build for submission.

14. References

1. "Unity User Manual: Basics of Game Development" by Unity Technologies
https://docs.unity3d.com/Manual/index.html
Accessed on **April 4, 2025**.

2. "Unity Learn: Official Unity Tutorials for Project Implementation" by Unity Technologies
https://learn.unity.com/
Accessed on **April 6, 2025**.

3. "How to Create a Tank Controller in Unity (Tutorial)" by GameDev Beginner
https://gamedevbeginner.com/how-to-make-a-tank-controller-in-unity/
Accessed on **April 7, 2025**.

4. "Unity Asset Store: 3D Models, Sounds, and Textures for Games" by Unity Technologies
https://assetstore.unity.com/
Accessed on **April 9, 2025**.

5. "Kenney Assets: Free 2D and 3D Game Assets" by Kenney
https://kenney.nl/assets
Accessed on **April 11, 2025**.

6. "Freesound: Collaborative Database of Creative Commons Licensed Sounds" by Freesound.org
https://freesound.org/
Accessed on **April 13, 2025**.

7. "Fundamentals of Game Design" by Andrew Rollings and Ernest Adams
https://www.pearson.com/en-us/subject-catalog/p/fundamentals-of-game-design/P200000003500/
Accessed on **April 15, 2025**.

8. "Creating Effective Sound Effects for Games" by Indie Game Developer Handbook
https://www.indiegamehandbook.com/sound-effects-guide
Accessed on **April 17, 2025**.

9. "Assistance through AI Tools like ChatGPT and DeepSeek AI for Code Debugging and Suggestions" by OpenAI and DeepSeek
https://chat.openai.com/
https://deepseek.com/
Accessed on **April 22, 2025**.

10. "The Art of Game Design: A Book of Lenses" by Jesse Schell
https://www.crcpress.com/The-Art-of-Game-Design-A-Book-of-Lenses/Schell/p/book/9780367331589
Accessed on **April 27, 2025**.