Project Specification Document

Project Title: Asteroid Madness

1. **Pranay Kelotra IMT2023563 Pranay.Kelotra@iiitb.ac.in**
2. **Rajdeep Saha IMT2023600 Rajdeep.Saha@iiitb.ac.in**
3. **Sahas Sangal IMT2023556 Sahas.Sangal@iiitb.ac.in**
4. **Sahil Singh IMT2023521 Sahil.Singh521@iiitb.ac.in**
5. **Yash Anand IMT2023624 Yash.Anand@iiitb.ac.in**
6. **Rajan Samar IMT2023616 Rajan.Samar@iiitb.ac.in**

Project Description:

The Project is a is a 2D game, single-player experience where the player controls multiple different kind of satellites to protect Earth from asteroids. The objective is to shoot down asteroids before they collide with Earth. The player can use different kind of missiles attached to satellites to destroy multiple kind of asteroids. The game will become increasingly difficult as the player score progresses. The players job is to direct the missiles along the path of the asteroids by using the cross hair.

Objectives:

1. **Core Gameplay (Single Player Mode):**
    a. Designing a game Mechanism for the Player to shoot down asteroids by launching missiles aimed along the path of asteroid.

b. Power-ups which can be unlocked and which can provide advantage to the player in saving the earth. **# POWERUPS ??**
c. Different kind of asteroids for the player to tackle differently.
d. Different kind of satellites for the player to choose while battling asteroids.
e. Various missiles having different effects on asteroids.
f. Game mechanism for ending the game if a asteroid collides with the earth **# EARTH TO HAVE LIFE ??**

2. **Game Progression:**

   #The game will feature different levels of increasing difficulty, with more asteroids and faster-moving ones as the game progresses.

   a. Game mechanism where the game become increasingly difficult as the players score increases.
      # Multiple difficulty settings (Easy, Medium, Hard) will affect the frequency, size, and speed of asteroids ??
   b. a game score based on how many asteroids player destroyed, and tracking the player's highest score.
      # Area where the missiles get automatically destroyed ?
      # A wave of asteroids or A BIG BIG ASTEROID ??
3. **UI and Visuals:**
   a. The game to include a simple and intuitive GUI, displaying information such as:
      i. Player score.
      ii. The graphics which display accessible satellites , asteroids and missiles
      iii. Cross Hair which displays where the user is currently aiming.
      iv. Cool-down period of satellites and other satellite features

Scope of the Project:

The primary focus of this project is to create a polished **single-player experience** where the player protects Earth by controlling a satellite.

Functional Requirements:

1. Core Game Mechanics :

- **Asteroid Spawning**:
  spawning different asteroids and their trajectory onto the screen
- **Missiles Shooting**:
  Shooting different missiles attached to the satellite and showing their trajectory onto the screen
- **Interactive GUI:** A game-screen where player is able to direct missiles (with help of a input device) onto asteroids and being able to see and identify different missiles satellites,asteroids and missile blasts
- **Collision Detection:**Detection when missiles collide with asteroids and when asteroids collide with Earth.
- **Score and Difficulty**:Players earn points for each asteroid destroyed and difficulty increases as score increases
- **Game Over Condition**:The game ends when an asteroid hits Earth, and the player's final score is displayed.

Non-Functional Requirements:

1. **Performance**:
   a. The game should run smoothly at a minimum of 60 frames per second (FPS) for a fluid gameplay experience.
   b. Efficient handling of collision detection and missile-asteroid interactions even at higher difficulty levels with numerous asteroids.
2. **User Experience**
   a. Game should be playable the player must not find it too difficult or too easy nor find it boring.
   b. Player must find the game exciting to play due to cool features and gameplay.

3. **User Interface (UI) Design**:

a. Simple and clear UI elements to display key game information (e.g., player score, asteroid count).
b. Smooth, responsive controls with immediate feedback when the player fires missiles or moves the satellite.

Technologies to Be Used:

- **Programming Languages**:
  - C++ and Java for core game logic and GUI.
- **Libraries/Frameworks**:

  ## WHICH LIBRARIES TO USE
  - **SDL2** or **SFML** (C++) for handling 2D graphics, input, and sound in the single-player mode.
  - **JavaFX** or **libGDX** (Java) for graphics and input handling if using Java.

- **Development Environment**:
  - **Visual Studio, IntelliJ** for servicing as IDE**.**
  - **Git** for version control.
  - Github for efficent sharing and hosting of code.

System Architecture:

1. Single-Player Game Architecture:

- **Game Loop**:
    - The game loop will handle asteroid spawning, satellite movement, collision detection, and missile firing.

    - Each frame, the game will update the positions of all objects (satellites, asteroids, missiles) and render the updated state to the screen.
- **Collision Handling**:
    - The game will continuously check for collisions between missiles and asteroids.
    - It will also check if any asteroids have collided with Earth, triggering the game-over condition.

3. Rendering and Input Handling:

- **Rendering**:
    - 2D rendering of Earth, satellites, asteroids, and missile trajectories.
    - Dynamic background indicating space with simple visual effects for explosions when asteroids are hit.
- **Input Handling**:
    - Keyboard input for satellite movement and firing.
    - Multiplayer input from multiple players, capturing both players' movements and actions.

Project Phases:

1. **Phase 1: Single-Player Core Development**
    a. Implement the core game mechanics (satellite movement, missile firing, asteroid spawning).
    b. Test collision detection and game-over conditions.
    c. Build and refine the UI to display player score, lives, and game state.
2. **Phase 2: Difficulty Levels and Asteroid Variety**
    a. Implement different asteroid types and difficulty levels (Easy, Medium, Hard).
    b. Add variety in missile types and asteroid behavior.
3. **Phase 3: Testing and Polishing**

    a. Thoroughly test the single-player mode, ensuring smooth gameplay and no major bugs.
    b. Refine performance and UI for a clean user experience.
4. **Phase 4: Local Multiplayer**
    a. Implement the client-server architecture over LAN for multiplayer functionality.
    b. Synchronize game state between players and ensure smooth communication.
    c. Test multiplayer performance and fix any synchronization issues.

Expected Outcomes:

- A fully functional, polished **single-player game** where the player controls a satellite to defend Earth from asteroids.
- A smooth and engaging gameplay experience, with increasing difficulty levels and a variety of asteroid and missile types.
- Optionally, **local multiplayer over LAN** as an additional feature, allowing two players to collaborate in protecting Earth from asteroids.