Seekers Test Plan for Milestone 1

CPSC 427 – Video Game Programming

# Team: Los Pollos Hermanos Gaming (LPHG)

Samuel Yiu (32494940)

Ahmed Al Busaidy (29200722)

Jakob Khalil (86176674)

Amir Goodarzvard Chegini (56851199)

Brendon Son (87271490)

## 1. Rendering

### 1.1. Textured Geometry

* Verify that all game entities (player, enemies, walls, trees) are rendered with appropriate textures.
* Check that there are no visual glitches or unexpected rendering artifacts.

### 1.2. Basic 2D Transformations

* Observe the player character and enemies to ensure they translate (move) correctly.
* Verify that entities rotate properly (e.g., player facing direction, enemy orientation).
* Check if any entities scale correctly.

### 1.3. Key-frame/State Interpolation

* Observe the player's movement and verify smooth interpolation between positions when dashing (click spacebar).
* Check enemy movements for smooth patrolling.
* Verify smooth rotation of entities when changing directions.

## 2. Gameplay

### 2.1. Keyboard/Mouse Control

* Test player movement using W, A, S, D keys.
  + Press and hold W to move the player character up.
  + Press and hold S to move the player character down.
  + Press and hold A to move the player character left.
  + Press and hold D to move the player character right.
  + Release keys (W, A, S, D) to stop the player's movement.
* Test player diagonal movement:
  + Press and hold W + A to move the player character diagonally up-left.
  + Press and hold W + D to move the player character diagonally up-right.
  + Press and hold S + A to move the player character diagonally down-left.
  + Press and hold S + D to move the player character diagonally down-right.
  + Release both keys to stop diagonal movement.
* Verify that the player rotates to face the mouse cursor (Only in 3D mode - see 4.3).
* Test shooting projectiles with left mouse click. Verify that projectiles are created and move toward the mouse cursor’s direction.
* Check if the spacebar triggers the dodge mechanic and verify the player's position is shifted in the direction of the movement key used.

### 2.2. Random/Coded Action

* Observe enemy behavior to ensure they exhibit some form of autonomous movement.

### 2.3. Game-space Boundaries

* Try to move the player beyond the edge of the map and verify that they are restricted from going out of bounds (refer to World.cpp, lines 44-70).
* Verify that the player cannot move outside the defined game area.
* Check if there are any glitches or unexpected behavior at the boundaries.
* Try to move the player through walls or obstacles and verify that they cannot overlap.

### 2.4. Collision Detection & Resolution

* Test collisions between:
  + Player and walls
  + Player and enemies
  + Player and trees
  + Projectiles and enemies
  + Projectiles and walls/trees
* Verify that entities don't overlap or pass through each other.
* Move the player towards an enemy and verify that they do not overlap. Player should push the enemy back.
* Aim at an enemy with the mouse cursor and shoot within attack range.
  + Verify that the projectile disappears on contact.
  + Verify that the enemy health bar decreases on contact.
  + Verify that the enemy disappears and the weapon is dropped when the health is depleted.

## 3. Stability

### 3.1. Frame Rate

* Play the game for at least 2 minutes, monitoring for consistent frame rate.
* Check for any noticeable lag or stuttering during gameplay.

### 3.2. Crash and Glitch Testing

* Perform rapid inputs and erratic movements.
* Test edge cases like rapid firing, quick direction changes, and colliding with multiple objects simultaneously.
* Play through different scenarios to ensure no crashes or unexpected behavior occurs.

## 4. Additional Features

### 4.1. Health System

* Verify that player and enemy health bars are visible and correctly positioned.
* Check if health bars update properly when damage is taken.
* Ensure entities are removed when their health reaches zero.
* Reported bug: Weapon and health bar incorrectly rotate around the player.

### 4.2. Weapon System

* Verify that weapons are visible and correctly positioned relative to entities.
* Reported bug: Weapon and health bar incorrectly rotate around the player.

### 4.3. 3D Mode

* Toggle between 2D and 3D modes using the 'Z' key.
* In 3D mode, verify correct camera positioning and perspective.
* Check if all game elements render correctly in both 2D and 3D modes.

### 4.4. Camera Controls

* Test camera rotation using 'Q' and 'E' keys.
  + Press and hold Q to rotate the camera counterclockwise.
  + Press and hold E to rotate the camera clockwise.
  + Release keys (Q, E) to stop camera rotation.
* Verify smooth camera movement and correct positioning relative to the player.

### 4.5. Environmental Elements

* Verify the correct placement and rendering of trees and walls (trees and enemies are randomized).
* Check if these elements properly obstruct movement and projectiles.

## 5. Asset Verification

### 5.1. Textures

* Verify that all required textures are loaded and applied correctly:
  + Player character
  + Enemies
  + Weapons
  + Projectiles
  + Walls
  + Trees
  + Ground/map texture

### 5.2. Sound Effects

* Verify that appropriate sound effects play for:
  + Shooting projectiles
  + Player movement
  + Dodging
  + Background music

## Notes for Testers:

* The game should run for at least 2 minutes without crashing or freezing.
* Report any visual glitches, clipping issues, or unexpected behaviors.
* Pay attention to the responsiveness of controls and the smoothness of animations.
* Verify that the game maintains a stable frame rate throughout testing.
* Check for consistent behavior across different playthroughs.