Test Plan: Ricochet Rage

CPSC 427 – Video Game Programming

Term: 2024W1

Instructor: Chris Kerslake

Team: Sigma Studios

Milestone: 1

Abhigyan Dabla (28266047)

Maysam Farahani (95167003)

Jason Liu (16585622)

William Cheng (86583648)

Sean Ford (42305409)

# Table of Contents

[**Table of Contents 2**](#_hhz74rcrd7i8)

[**Setup 3**](#_jb22w985ub07)

[**Controls 4**](#_e1a90kt0jg0d)

[**Features 5**](#_t40s4brq4uy)

[Rendering 5](#_jwztvxs4ic3r)

[Gameplay 5](#_74wdcx226jq9)

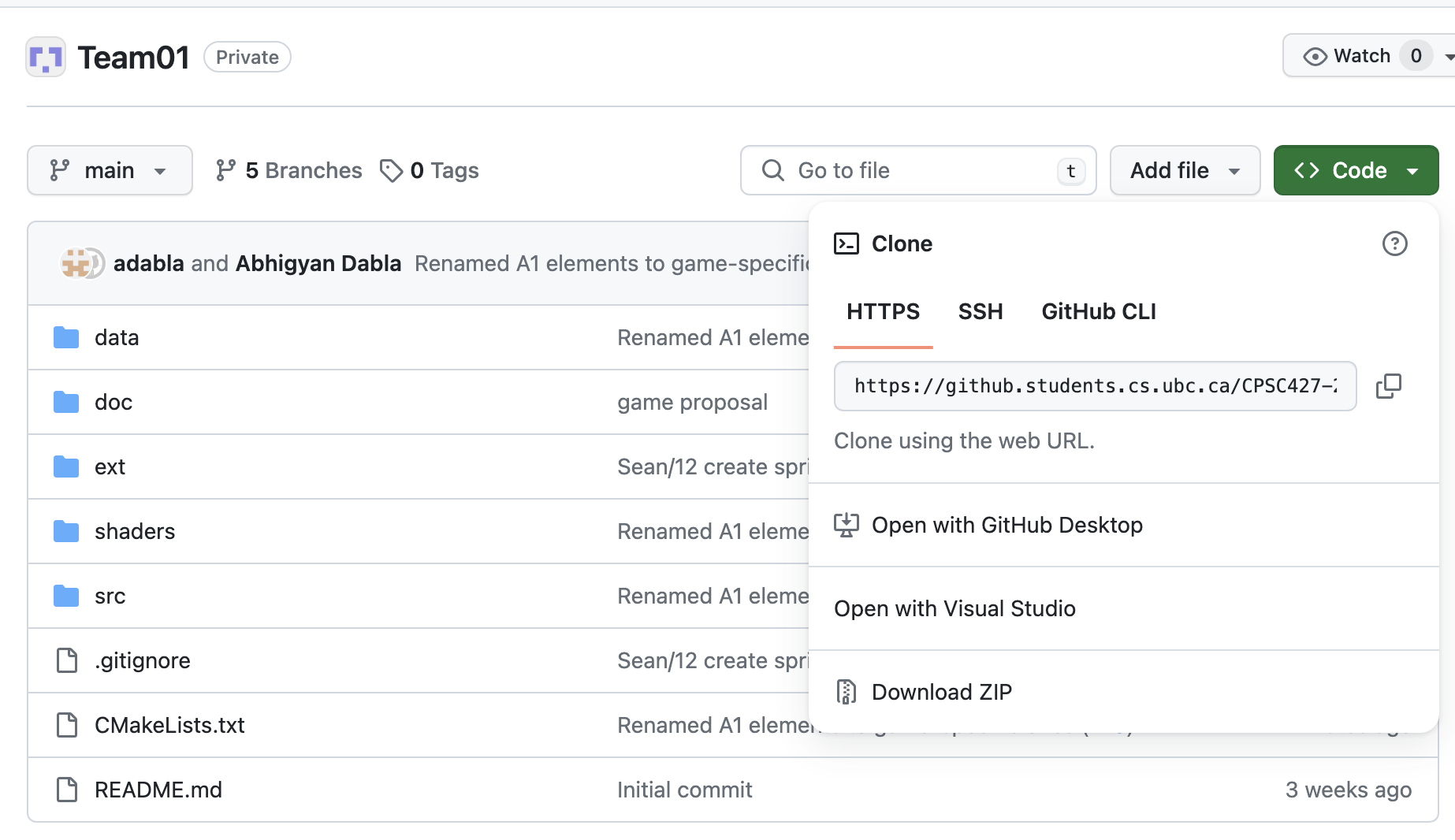
[Creative 6](#_rlj6us8aq1op)

# 

# Setup

In order to test *Ricochet Rage*, please follow the steps below to set up your development environment and run the game:

1. Set up your development environment to be compatible with C++ 14, CMake, and OpenGL if you have not already. Instructions for environment setup can be found on Canvas [here](https://canvas.ubc.ca/courses/147789/pages/tutorial-02-dev-env-setup?module_item_id=7236387).
2. Next, clone the GitHub repository to your local machine or download a copy as a ZIP file, then uncompress on your machine.



1. After successfully downloading the codebase to your local machine, open a terminal and create a new directory within the project called build/ by running mkdir build
2. Change into the directory by running the command cd build
3. Generate build files by running the command cmake ..
4. Compile and build the project by running the command cmake --build .
5. Finally, run the game with the executable by running ./ricochet-rage

# 

# Controls

The following key bindings are currently implemented for *Ricochet Rage*:

* **W**: Move up
* **A**: Move left
* **S**: Move down
* **D**: Move right
* **R**: Respawn
* **Space**: Dash
* **Left mouse click**: Shoot
* **Escape**: Exit game

Additionally, players can aim using mouse controls for more precise targeting.

# Features

The following sections outline the required and creative features implemented for Milestone 1. Use the descriptions below to test and verify each feature.

## Rendering

**Textured Geometry**

We implemented textured geometry using PNG sprites and shapes. The player is represented by a blue circle, while the enemy is a red circle. Textured PNGs are used for the game walls and bullets, which change color after ricocheting off walls.

**Basic 2D Transformations**

Players can move their character using WASD keys, which apply translation. This can be tested by moving around the game room with the WASD keys.

Projectiles bounce off walls using reflection. Test this by shooting projectiles with the left mouse button; they will reflect off walls twice before disappearing unless they hit a character.

**Key-frame/State Interpolation**

Interpolation is used for player movement, shooting projectiles, and dashing. Test movement using the WASD keys, shooting with the left mouse button, and dashing with the space key. The movement and shooting uses linear interpolation, while the dash uses non-linear interpolation.

## Gameplay

**Keyboard/Mouse Controls**

The player (blue circle) is controlled via keyboard and mouse inputs. Test all key bindings as listed in the [Controls](#_e1a90kt0jg0d) section.

**Random/Coded Enemy Actions**

The enemy (red circle) uses AI to track the player and fire projectiles when within range. Test this by moving the player closer to the enemy; the enemy will pursue, avoid obstacles, and shoot projectiles once the player is within range.

**Game-Space Boundaries**

Players and enemies cannot move beyond the visible room boundaries or through the obstacle in the center. Test this by attempting to move the player or enemy out of bounds or over obstacles.

**Collision Detection & Resolution**

We implemented collision detection for characters and projectiles to stay within game-defined boundaries. Test by moving the player toward walls and obstacles (using WASD) and shooting projectiles (left mouse button) to confirm boundaries are respected.

Damage detection is triggered when projectiles hit characters. Test by allowing the enemy to hit the player or by shooting the enemy. Characters will respawn or die as expected, and the game can be reset using the "R" key.

## Creative

**Ricochet Mechanic (Basic Feature #1)**

Projectiles bounce off walls using reflection. Test by shooting projectiles with the left mouse button and observing how they ricochet twice before disappearing unless they hit a character.

**Dashing Ability (Basic Feature #2)**

Players can dash to evade enemies. Test this by pressing the space key, which moves the player towards the direction of the mouse cursor.

**Enemy AI/Pathfinding (Basic Feature #3)**

The enemy uses pathfinding to pursue the player and avoid obstacles. Test by moving the player into the enemy’s range, triggering pursuit and projectile attacks.