Test Plan: Ricochet Rage

CPSC 427 – Video Game Programming

Term: 2024W1

Instructor: Chris Kerslake

Team: Sigma Studios

Milestone: 2

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)

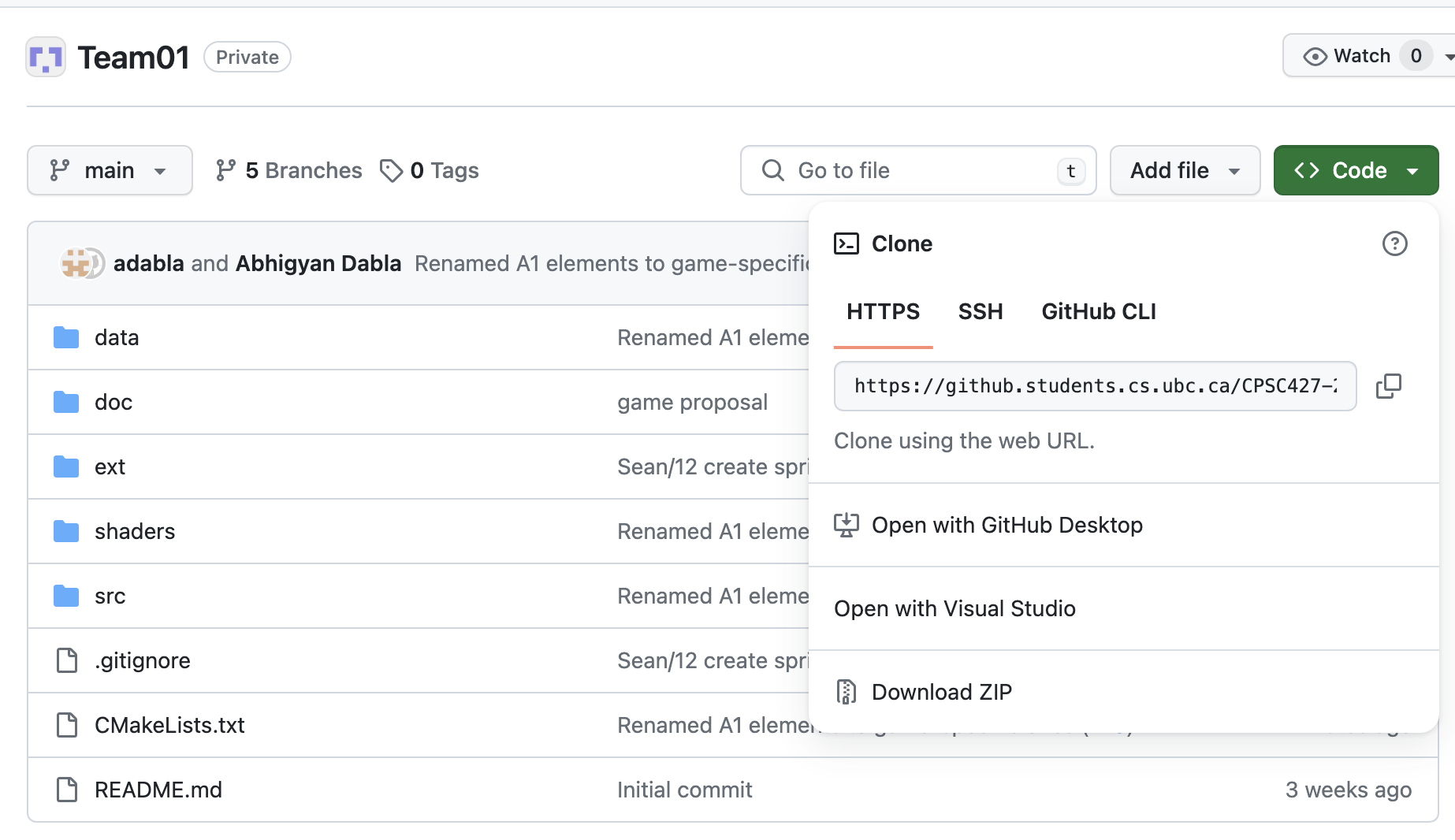
[Creative 6](#_rlj6us8aq1op)

# 

# 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
* **Space**: Dash
* **Left mouse click**: Shoot
* **Escape**: Pause Game

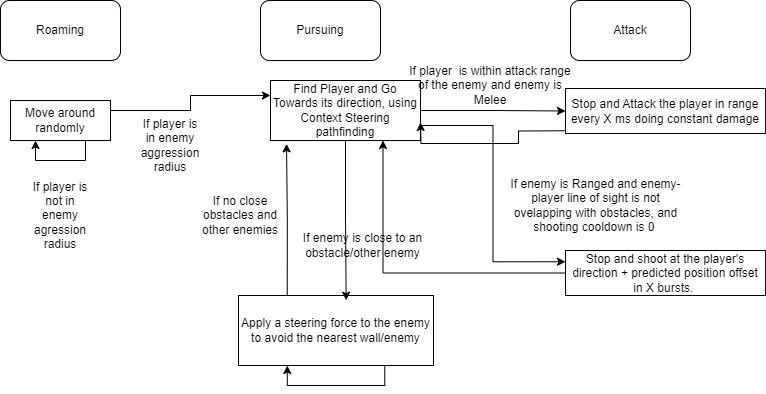
Additionally, players can aim using mouse controls for more precise targeting.

# Features

The following sections outline the required and creative features implemented for Milestone 2. Use the descriptions below to test and verify each feature.

## Improved Gameplay

**Game AI**



For game AI, we separated the enemy into 3 states: *roaming*, *pursuing*, and *attack*. When pursuing the enemy uses a Context Steering pathfinding algorithm. If the enemy is of type melee attack, it will attack the player in range. If the enemy is ranged, it will do a line of sight check to see if you have a clear shot at the player and shoot bursts of bullets.

*To test these features, ensure that all the states are reachable and resolve after attacks. Test melee and ranged attacks separately and ensure that it resolves to the pursuing state eventually. Test if the steering force is sufficient to steer away from the obstacles and continue the chase.*

**Animation**

Animations for players and enemies are implemented using sprite sheets. When you play the game, you’ll see animations for both players and enemies during movement, with a stationary pose when they are not moving.

*To test these features, play the game and observe the updated visuals.*

**Assets**

This milestone introduces enhanced graphics, including a new background, redesigned walls, characters, power-ups, and more. Pillars in the center of the room add dynamic gameplay as projectiles can bounce off them.

*To test these features, play the game and observe the updated visuals.*

**Mesh-Based Collision Detection & Resolution**

Mesh-box collisions are implemented between the mesh-based projectiles and many sprites in our game, such as players, enemies, and walls.

*To test this feature, shoot projectiles towards the walls and enemies to observe the precise collision detection.*

**Help**

A new tutorial window is available to guide players through movement, combat, and controls, with basic game instructions included.

*To test this feature, launch the game and select “Tutorial” from the main menu.*

**FPS Counter**

The current frame rate is displayed in the window’s title bar.

*To test this feature, start the game and check the FPS counter at the top of the game window.*

## Creative

**Serialization (Basic Feature #1)**We implemented a basic save feature that records the game state to a text file when the game closes, allowing it to be resumed next time.  
*To test this feature, exit the game using “Esc”, then reopen it and select “Resume Game” to continue from where you left off.*

**Audio Feedback (Basic Feature #2)**To enhance player feedback, sound effects have been added for shooting projectiles, enemy death, and player death.  
*To test this feature, engage in combat both by attacking and being killed.*