By: Theodore Chu, Justin Kyle Torres, and Alejandro Zapata

5.1 Introduction

K’two is a 1-4 player, browser-based, online multiplayer, class-based, zombie wave survival game. It is meant to carry on the legacy of a previous LMU capstone project K’tah by showcasing progression of game development tools and resources and our capabilities as developers. Networking, graphics pipelines, game design, are all technical problems we are eager to solve together as a team utilizing the four years of multifaceted education we have garnered from LMU. The remainder of this document is structured as follows: Section 5.2 functional requirements, Section 5.3 performance requirements, Section 5.4 environment requirements.

5.2 Functional Requirements

5.2.1 Server Lobby

A apparatus for players to look for other players to match up with.

5.2.1.1 Players shall be able to look for other players to match up with.

5.2.1.2 Players shall be able to create rooms that other players can join.

5.2.1.3 The server shall group players in the same room and redirect them to the same server or process to start gameplay

5.2.2 Encounters

Encounters represent the in-game environment players will be in as they attempt to survive enough waves of zombies to pass the encounter. They will use their abilities and teamwork to circumvent the hoards long enough to succeed.

5.2.2.1 Players shall use their abilities to interact with each other and zombies.

5.2.2.2 Players shall see the same entities on screen as each other.

5.2.2.3 Players shall see the abilities of other players being used.

5.2.2.4 Players shall get hurt when attacked by a zombie.

5.2.2.5 Players shall get hurt when attacked by a zombie.

5.2.2.6 Players shall get rewarded as a group a currency based on zombies killed and based on how far the group has survived so far.

5.2.3 Map Traversal

Before an encounter begins, players can traverse a node-based map system. Where they can only move forward and visit an empty node, a shop node, an exit node, or an encounter node.

5.2.3.1 All players shall see the same map.

5.2.3.2 Players shall be able to control which node they visit next.

5.2.3.3 The traversal shall be controlled by a host or “leader” .

- This will be given to whoever created the original room in the server lobby.

5.2.3.4 Players shall not be able to revisit previous nodes, as they can only move forward.

5.2.3.5 The appropriate scene shall be triggered depending on the node visited.

5.2.4 Shop

When visiting a shop node, a screen will pop-up, showing the available upgrades and allowing players to spend currency earned from Encounters to purchase these upgrades. A purchase will deduct

5.2.4.1 Only the host player shall be able to control during the shop screen.

- This will introduce a sort of metagame where the host player can choose whether to or not listen to the appeals of his team.

5.2.4.2 Purchases shall deduct the appropriate amount from the group’s shared currency.

5.2.4.3 Purchases shall be “upgrades” to the players according to the description given by the shop.

5.2.4.4 The shop shall have a progression system based on what the players have already bought and how far they’ve survived already.

5.2.4.5 The shop shall show appropriate options based on the classes picked.

5.2.5 Class Based Abilities

Each class has their own set of abilities specific for the player using the class. They are designed to work together and have synergies with each other.

5.2.5.1 At a minimum, there shall be 5 different classes.

5.2.5.2 Each class shall feel unique and have a defined role.

5.2.5.3 Each class shall be graphically unique.

5.2.5.4 Each class should have synergistic properties for the overall group.

5.2.5.5 Each player shall not pick a class that has already been picked.

5.2.6 Zombies

Zombies only show up in encounters and spawn at the edge of a map. They will have a basic AI in which they path a straight line to nearest (or random) player.

5.2.6.1 Zombies shall path to the nearest (or random) player.

5.2.6.2 Zombies shall attack any entity they collide with (player or wall or trap).

5.2.6.3 Zombies shall die (disappear from the game) if they run out of health.

5.2.6.4 Zombies shall receive damage accordingly.

5.2.6.5 Zombies shall receive the debuffs/buffs when triggered by players’ abilities.

5.2.7 Tracked High-scores

A central conceit of K’two is the ability to earn high scores that will be immortalized forever in the Keck Lab servers. The score a group of players earn is determined by how many encounters they survive.

5.2.7.1 The players’ score will be updated after an encounter.

5.2.7.2 When a team completely loses, they shall be taken to the high score screen where they submit their name and high score to the server.

5.2.7.3 There shall be a server keeping track of high scores and will display the top 15 scores.

- It should display the name, date, and score from a group.
- The list will be numbered.
- This will be persistent and viewable by players from the server lobby.

5.3 Performance Requirements

5.3.1 Boot-up

5.3.1.1 Boot-up may take 20-30 seconds as the user has to download the entire game on startup since it is a browser-based game.

5.3.2 High-Score Submission

5.3.2.1 Submitting a high-score at the end of a playthrough should take no longer than 15 seconds for the server to submit a persistent score to the high score list.

5.3.3 In-game Encounter

5.3.3.1 Transitioning to an in-game encounter should take no more than 30 seconds.

- Each client needs to load assets, and the server needs to ensure all clients are connected before commencing the encounter.
- Lag or delay should be minimized where possible to minimize experience detracting.

5.3.4 Server Lobby

5.3.3.1 It should take less than 10 seconds to load the server list.

5.3.3.2 It should take less than 15 seconds to begin hosting a room.

5.3.3.3 It should take less than 10 seconds to connect to a host.

5.3.3.4 It should take less than 10 seconds for the server to commence a game after all players have readied up.

5.4 Environment Requirements

5.4.1 Environment Execution Requirements

The following are the hardware requirements for K’two:

Processor	1.3GHz dual-core Intel Core i5
Hard Drive Space	N/A (Browser-based application)
Ram	4GB
Display	1280x720
Sound Card	Optional
Graphics Card	Intel HD Graphics 4000

The following are software requirements for K’two:

Operating System	Windows / MacOSX / Linux
Browser	Chrome / Firefox
Graphics Plugin	WebGL (for the browser)

5.4.2 Development Environment Requirements

The following are the hardware requirements for K’two:

Processor	2.4GHz dual-core Intel Core i5
Hard Drive Space	N/A (Browser-based application)
Ram	4GB
Display	1920x1080
Sound Card	Required
Graphics Card	Intel HD Graphics 4000 (Discrete Graphics recommended)

The following are software requirements for K’two:

Operating System	Windows / MacOSX / Linux
Browser	Chrome / Firefox
Graphics Plugin	WebGL (for the browser)
Development Editor	Unity 2018
3D Modeling/Animation Software	Maya AND Blender