**Software Requirements and Design Document**

**For**

**Group <4>**

Version 1.0

**Authors**:

Matthew B

John R

Mina A

Ethan P

Jared G

# Overview (5 points)

A 16-bit 2D platformer where the player is a monkey called Primate Ape, traveling its way through multiple levels, defeating lions and liberating levels by traveling to the end of each one. The enemies are lions.

# 2. Functional Requirements (10 points)

# The enemies walk around the level. (high)

# The enemies cause the health bar to go down when touched from the side. (high)

# There is an always-visible health bar that reflects the health level of the main character (high)

# Lions attack by shooting a bow or using a sword. (high)

# The character is controlled through user input of the keyboard (high)

# The character can jump. (high)

# The character has a static health bar at the top left of the screen. (high)

# Enemies can die. (high)

# The main character dies when the health bar runs out. (high)

# The level restarts every time the main character dies. (high)

# The camera moves as the character moves. (high)

# There's collision detection between the character and the levels. (high)

# The next level must be unlocked at the end of a level, excluding the third level. (high)

# The default ability is being able to jump on enemies. (high)

# The user can get new abilities after beating each level. (high)

# The player can jump on enemies to decrease enemy health. (high)

# There are enemies in each level that have their own behaviors. (high)

# Enemies aim towards the player. (high)

# Once life energy is collected, the health bar can be refilled. (high)

# Once a level is finished, the health bar is refilled. (high)

# Once the health bar runs out, the user starts from the beginning of a level. (high)

# Each level you beat in this platformer unlocks a new level. (medium)

# Collecting health allows the player to increase health. (medium)

# In order to beat a level, the player must interact with a chest at the end of the level. (medium)

# A text dialog pops up after beating each level, stating that the player has unlocked a new weapon. (medium)

# Several monkeys in a cage must be freed at the end of the level. (medium)

# Each level has a different design: a jungle, a desert, and a massive fortress. (medium)

# Defeating the first level unlocks a punch attack. (medium)

# Enemies only shoot projectiles when you see them on the screen. (medium)

# When an enemy is defeated, there’s a 25% chance of it dropping life energy. (medium)

# There are three levels in the game. (low)

# There is a final boss at the end of the game. (low)

# There will be a pause screen, which will pause everything in the game. (low)

# The pause screen will allow the user to either resume or exit the game. (low)

# The game will start by displaying text bubbles that describe the storyline (low)

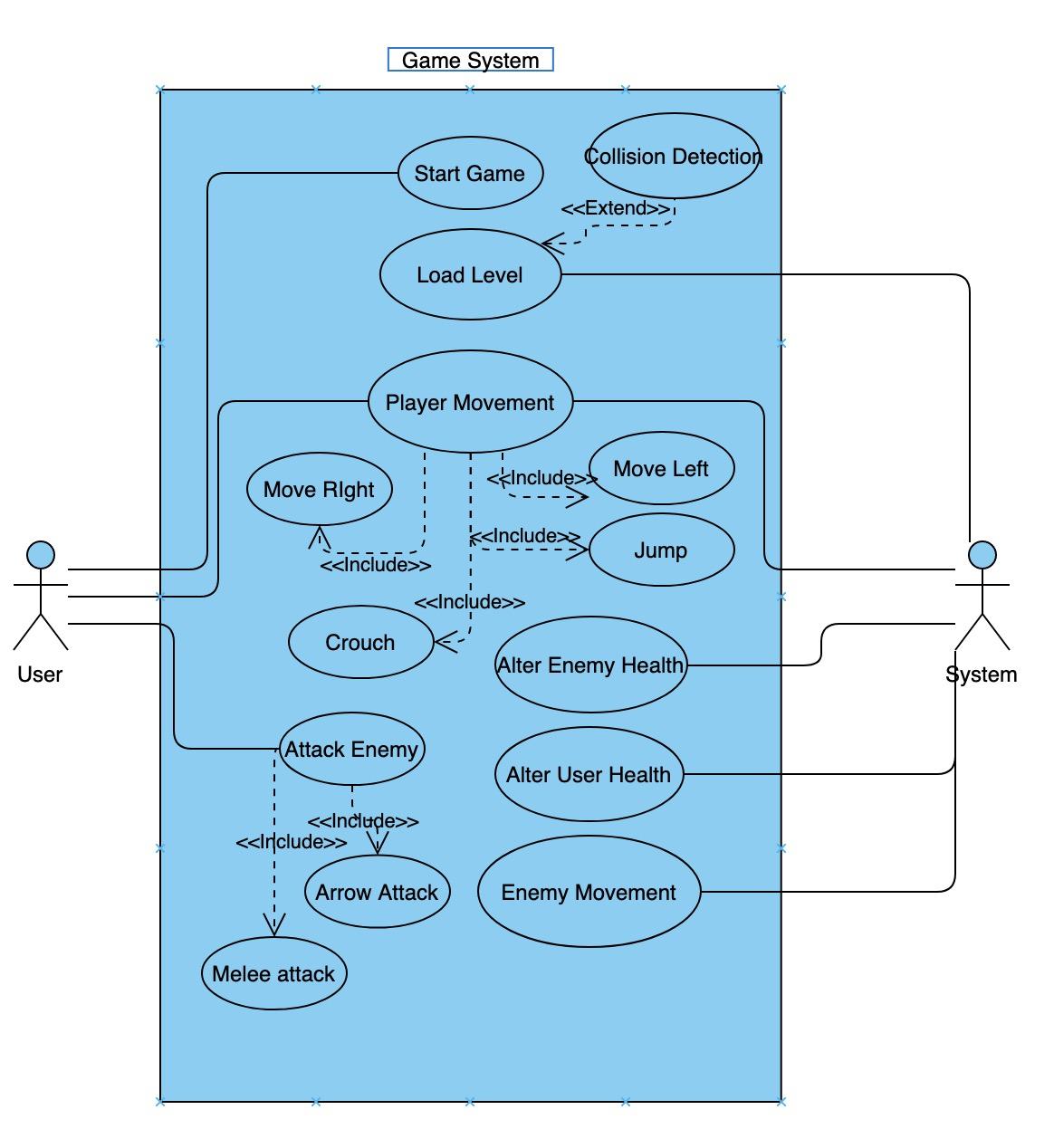
# Defeating the second level unlocks a gun. (low)

# Projectiles from the gun cannot escape the view of the player camera. (low)

# Non-functional Requirements (10 points)

1. The game should start up in less than 5 seconds.
2. Loading between levels should take less than 5 seconds.
3. There should be no noticeable input delay (>30ms).
4. The game will be well optimized for the user.
5. The game will not crash unexpectedly.
6. The game will function on Windows and MacOS operating systems.
7. The health bar will accurately reflect the player's health.
8. Damage done to enemies will stay consistent between attacks ex: (jump deals one damage).

# Use Case Diagram (10 points)



**Start Game -** This will occur when the user boots up the game and provides instructions to the system to load the level.

**Load Level -** This is the first action of the system when the user starts the game and creates the world that the player will be initialized within.

**Collision Detection -** this use case is extended by the Load Level because this use case provides physics to define how the lions, user, and weapons interact within the world.

**Player Movement -** This is an integral part of the player experience and defines how movement within the game is conducted. This use case includes Move Left, Move Right, Crouch, and Jump.

**Attack Enemy -** The user has the ability to attack the enemy using several attacks

**Move Right -** The user will move to the right by pressing the right button or the ‘D’ button.

**Move Left -** The user will move to the left by pressing the left button or the ‘A’ button.

**Jump** - The user will travel in the air and descend accordingly by pressing the ‘W’ button or the up button.

**Crouch -** The user will crouch downward by pressing the ‘S’ button or the down button. This will allow the user to travel under certain platforms that they could not travel under otherwise.

**Alter Enemy Health -** This use case is necessary in order to analyze when an enemy is damaged and how much health they lose for every attack from the user

**Alter User Health -** This use case is necessary in order to decrease and increase the users health depending on if they are attacked by an enemy or enter a new level

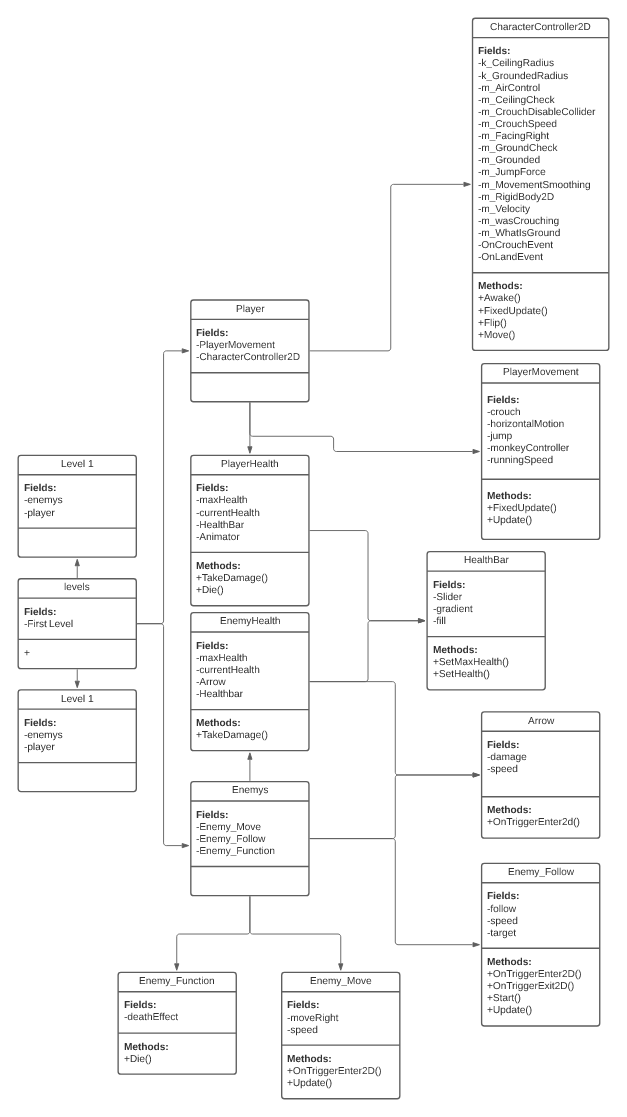
**Enemy Movement -** This tells the enemy how it should move within the level currently there are two different types of movement for the enemies. The first type allows the enemy to move back and forth and the second allows the enemy to follow the player when in range.

**Arrow Attack -** The user shoots an arrow. It hits the enemy if the arrow gets into the enemy’s hitbox, otherwise it does not hit.

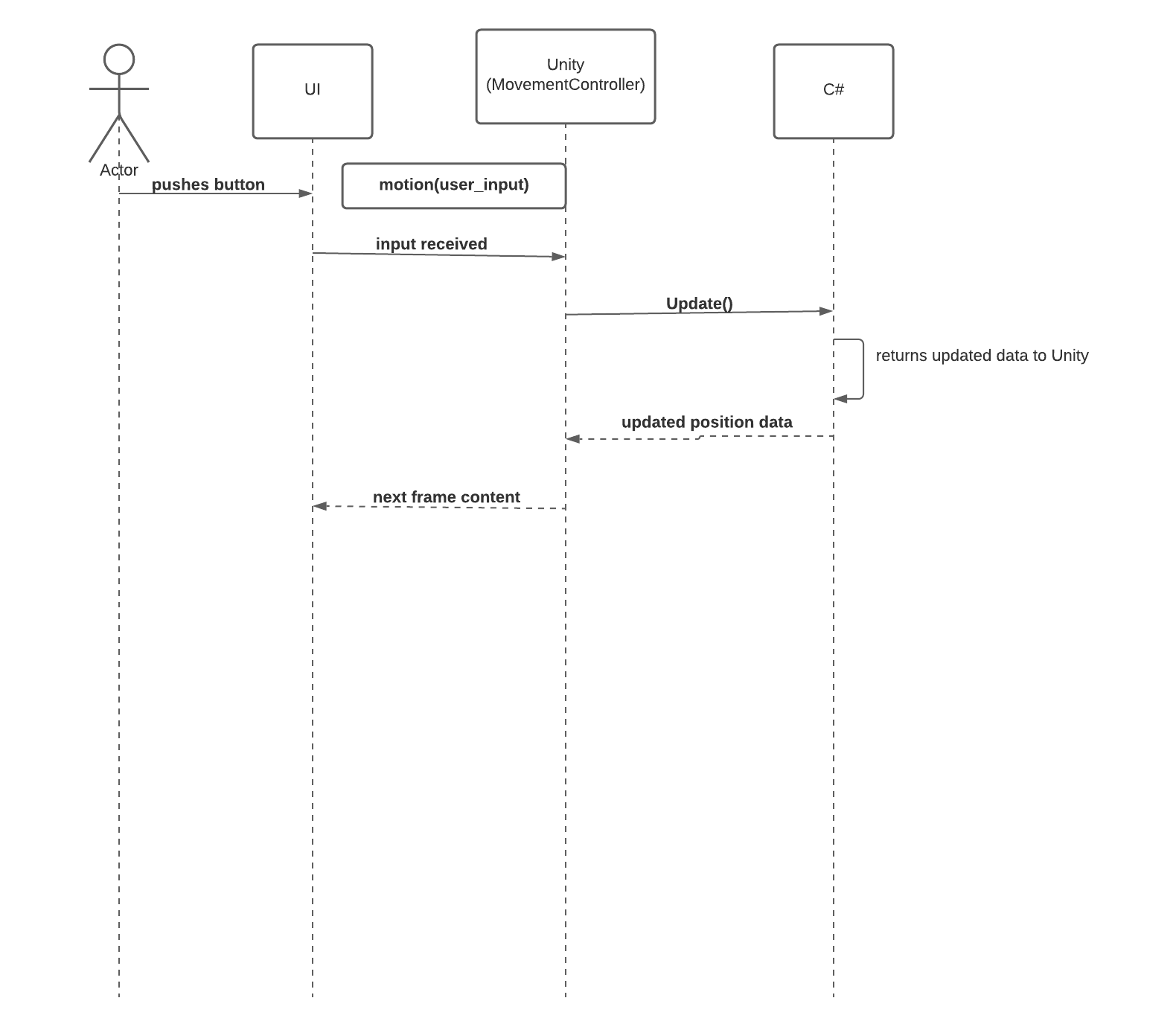
**Melee Attack** - The user punches the enemy. It hits if it lands in the enemy’s hitbox.

# Class Diagram and/or Sequence Diagrams (15 points)

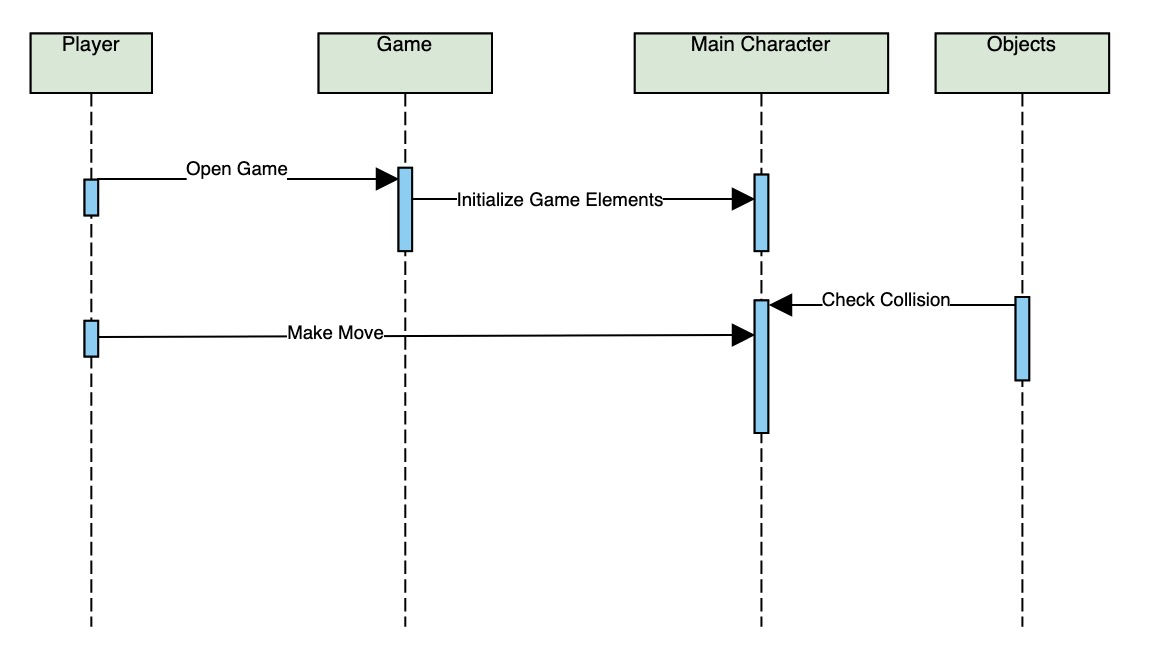
Class Diagram:



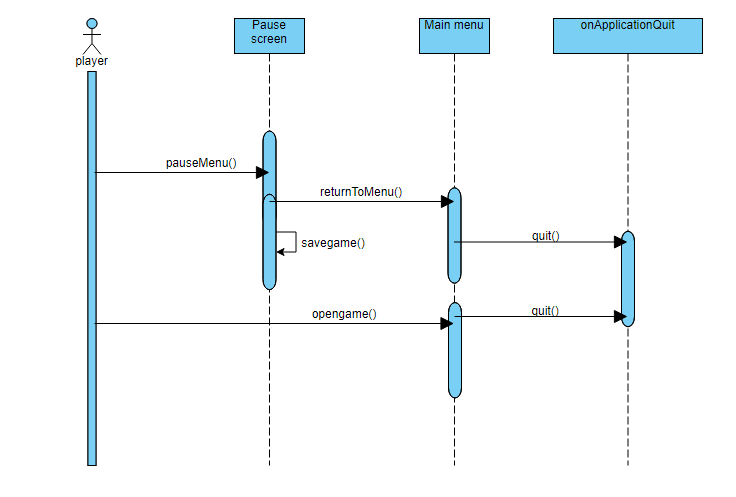
Control Character Use Case:

**

Start Game Use Case:

**

End Game Use Case:



# Operating Environment (5 points)

The software will operate on a desktop or laptop environment. It will be able to run on many other platforms, but for our purposes it will be run on a MacOS or windows machine. The software requires the Unity launcher for it to run.

# Assumptions and Dependencies (5 points)

Assumptions:

* The Unity engine functions properly and doesn’t cause unexpected issues during the development process
* The requirements are able to be implemented within the timeframe of the project deadline

Dependencies:

* The user is able to run the program on their local machine using the Unity game engine interface without performance issues
* The program runs on Unity version 2021.1.0b6