Software Requirements and Design Document

For

Group - Killer Among Us

Version 2.0

Authors:

Supriya Palli

Thuyvan Vulam

Joseph Ene

Anika Patel

Alec Amico

1. Overview (5 points)

We are proposing to create a mobile app platform-based game with a storyline integrated in it. The storyline is about a boy who takes matters into his own hands to find the killer of his mother. During the levels, he faces obstacles and is given clues on the journey towards the killers from ghosts who have passed due to the mysterious incident. At the end of the game, the player will find out the true identity of the killer.

2. Functional Requirements (10 points)

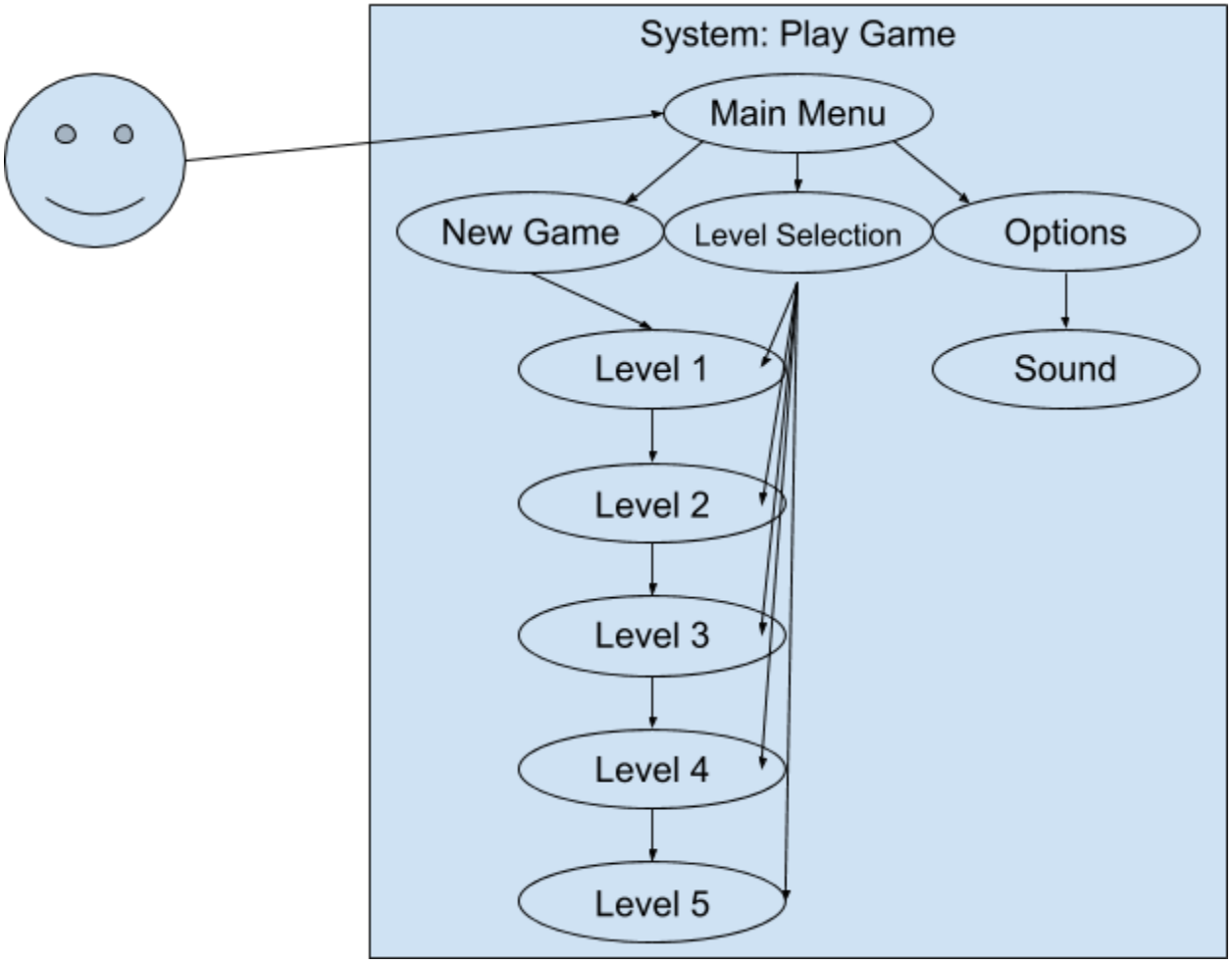
1. Player presses left, right, or jump button -> Character should move to the left, right, or jump (High)
2. When the character hits enemy characters, weapons, or falls off platforms -> Character should respawn to beginning of level/back to checkpoint (Medium)
3. Character moves near ghosts -> Ghost dialogue pops up and the player learns more about the story (Low)
4. Player reaches end of platform level by traveling through the door at the end of the level ->Takes player to dialogue portion of the game (High)
5. Player chooses to start game/choose level selection -> Takes player to the specific game level (Medium)
6. Player presses options menu -> Player can adjust the volume (Low)
7. Player reaches a checkpoint -> Flag should change colors from red to green (Low)
8. Player collides with an item -> item should be “picked up” and disappear from the screen, should be saved into a variable or trigger an action based on object tag (Medium)
9. Player has or does not have item -> triggers or fails to trigger an action based on item possession (Medium)
10. Player presses a pause menu input button -> Player can choose to select resume, restart, main menu, settings, or quit (Medium)
11. Pause Menu Resume button -> Player can resume the game. Unfreezing player and enemy movements, and allow user input to go through again. (Medium)
12. Pause Menu Restart button -> Level should restart with character being in beginning position for the level. (Medium)
13. Player reaches a dialogue scene -> Player should be able to choose options for which dialogue they want to respond with (High)
14. Dialogue scene ends -> Dialogue scene should shift back to level selection screen (High)

15. Player reaches falling hazard portion -> player should die if hit by falling hazard (Medium)
16. Player encounters enemy -> player should die if touching enemy (Medium)
17. Player collides with ladder -> player should be able to move up and down ladder (High)

3. Non-functional Requirements (10 points)

1. To implement responsive user input, we have to tune things like run, jump, and speed, or how fast gravity affects the player object.
2. To avoid collisions between the player and the platform, individual tiles which make up a map had to be treated as one large tile in the foreground. This is necessary because otherwise the player can clip into each individual tile in the scene.
3. UI & Controls needs to be iOS compatible, Standard Assets - CrossPlatformInput is essential.
4. Minimal frame rate needs to be at least 30 fps in order to guarantee a positive game experience.
5. Needs to be small enough to not take up too much RAM on an iOS mobile device.

4. Use Case Diagram (10 points)

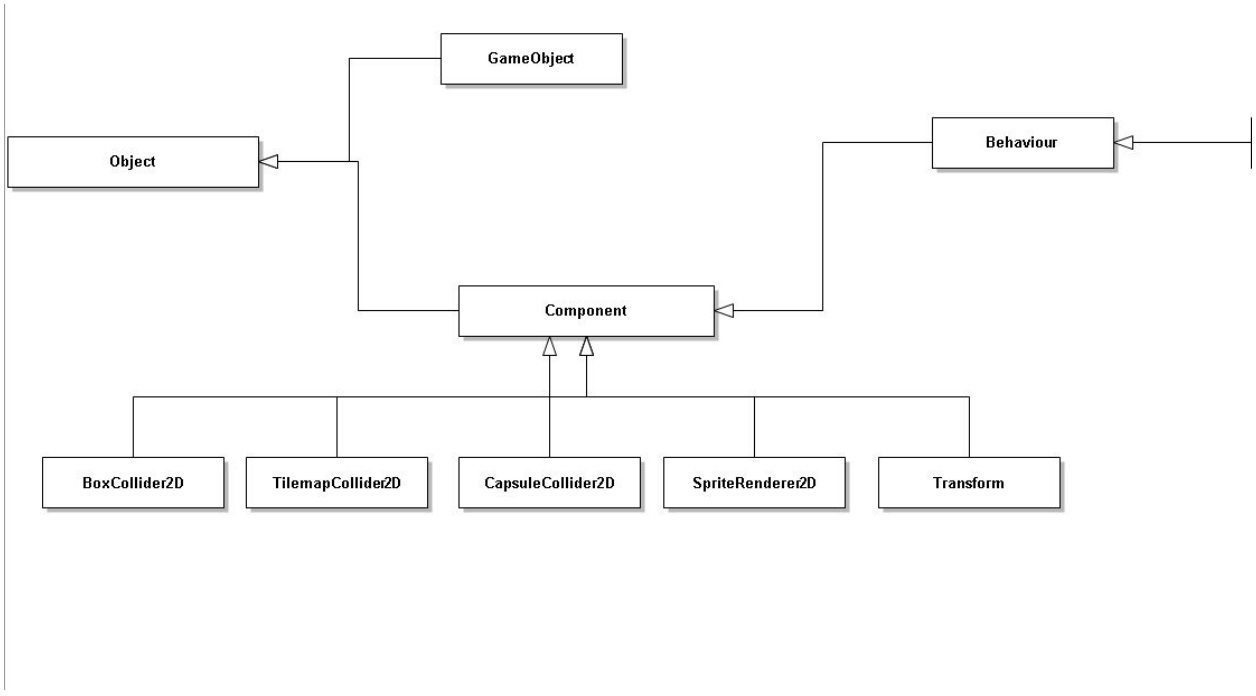


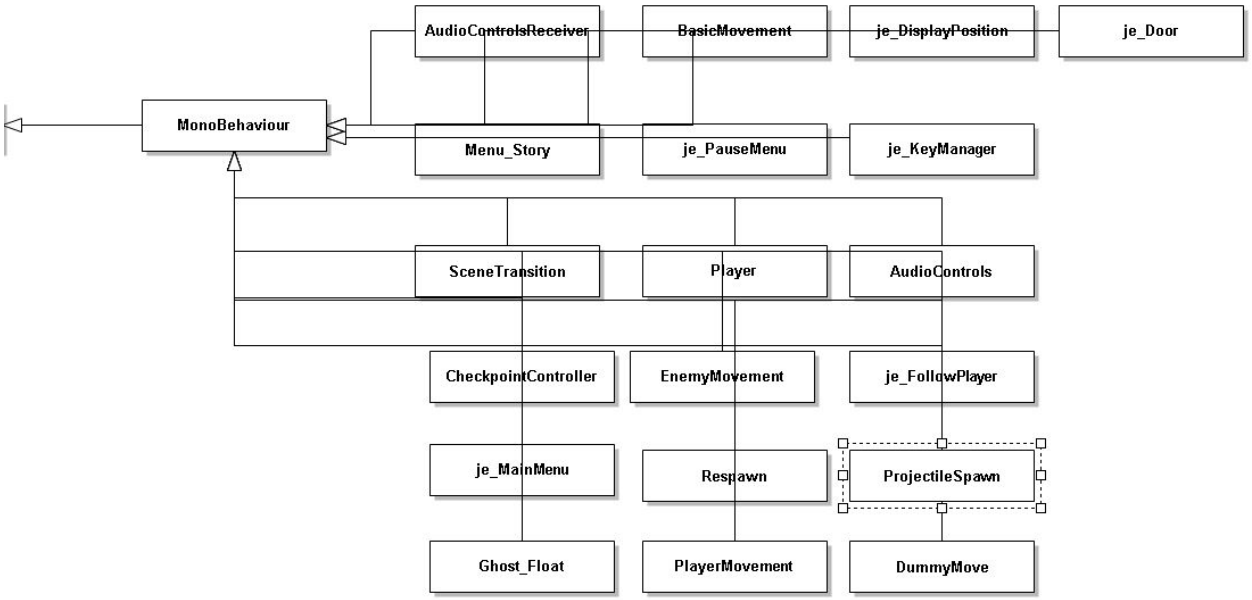
Textual Description of Use Case for “Play Game”	
Actors	Player
Preconditions	Player must be able to successfully open/load the game.
Normal Flow/Path	Player goes through levels linearly and completing levels unlocks them.

Postconditions	Finishing the game unlocks all levels.

5. Class Diagram and/or Sequence Diagrams (15 points): Class Diagram below

** MonoBehaviour inherits from Behaviour **





6. Operating Environment (5 points)

The operating environment will be iOS on a mobile platform. It will be working with the Unity 2019.3.15f version and the current iOS version 13.5.1, as the build of the game will be on the latest iOS version, therefore it must be a version of Unity that is compatible with iOS and keeps up compatibility if the iOS system is to be updated. Alternatively, the iOS version must be compatible with the version of Unity being used.

7. Assumptions and Dependencies (5 points)

There are online assets that are being used in the game, so for future development purposes, if those resources were no longer available or licensing permissions were changed, then it would affect the animation portion. The software will be dependent on Unity working properly with iOS, due to this dependency, updates within the iOS system and Unity versions must be taken into account for compatibility with one another. An assumed factor is a technology based assumption that we will be able to implement all our features such as dialogue box interaction, enemy and obstacles, etc. with the tech available with Unity; and if our assumptions are wrong we may have to change some plans. Another assumption is time based, that we would be able to make a base level for this increment so we have an example to go off of in future increments, and if this is not done, it could set us back time wise because we would lack a visual and basic components/scripts to proceed to make additional levels and need more time to add them. Additionally, we had certain design based assumptions that we would decide upon now in order to proceed with the building process, such as the storyline and level setup, and if those aspects were not decided upon, we would have been behind with no set idea of how we would want to proceed. An assumption around technology was also that we would have Unity and the GitHub extention’s cooperation and it would be smooth running for the most part, however we ran into plenty of challenges on both ends that were not anticipated and affected timing of some plans.