TEAM UNITY PROTOTYPE 3 REPORT

EMRE SEZER

BİLAL YALÇINKAYA

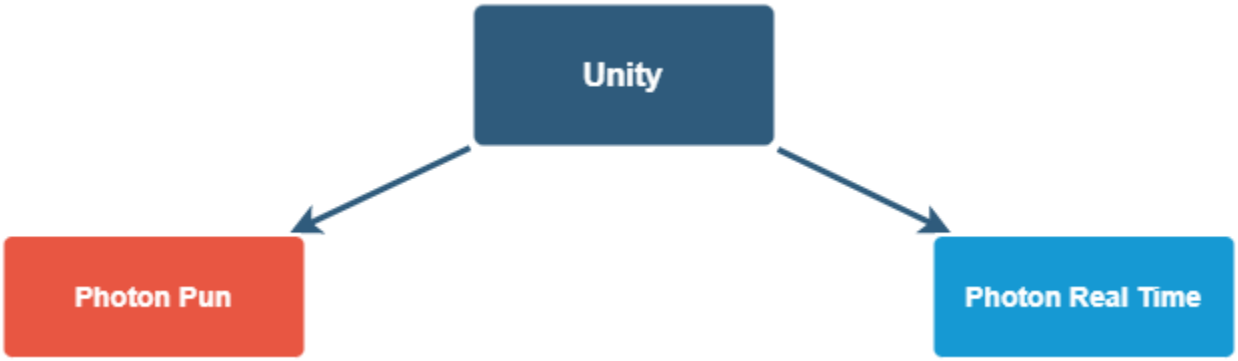
HAKAN KESKİN

AHMET OKUR

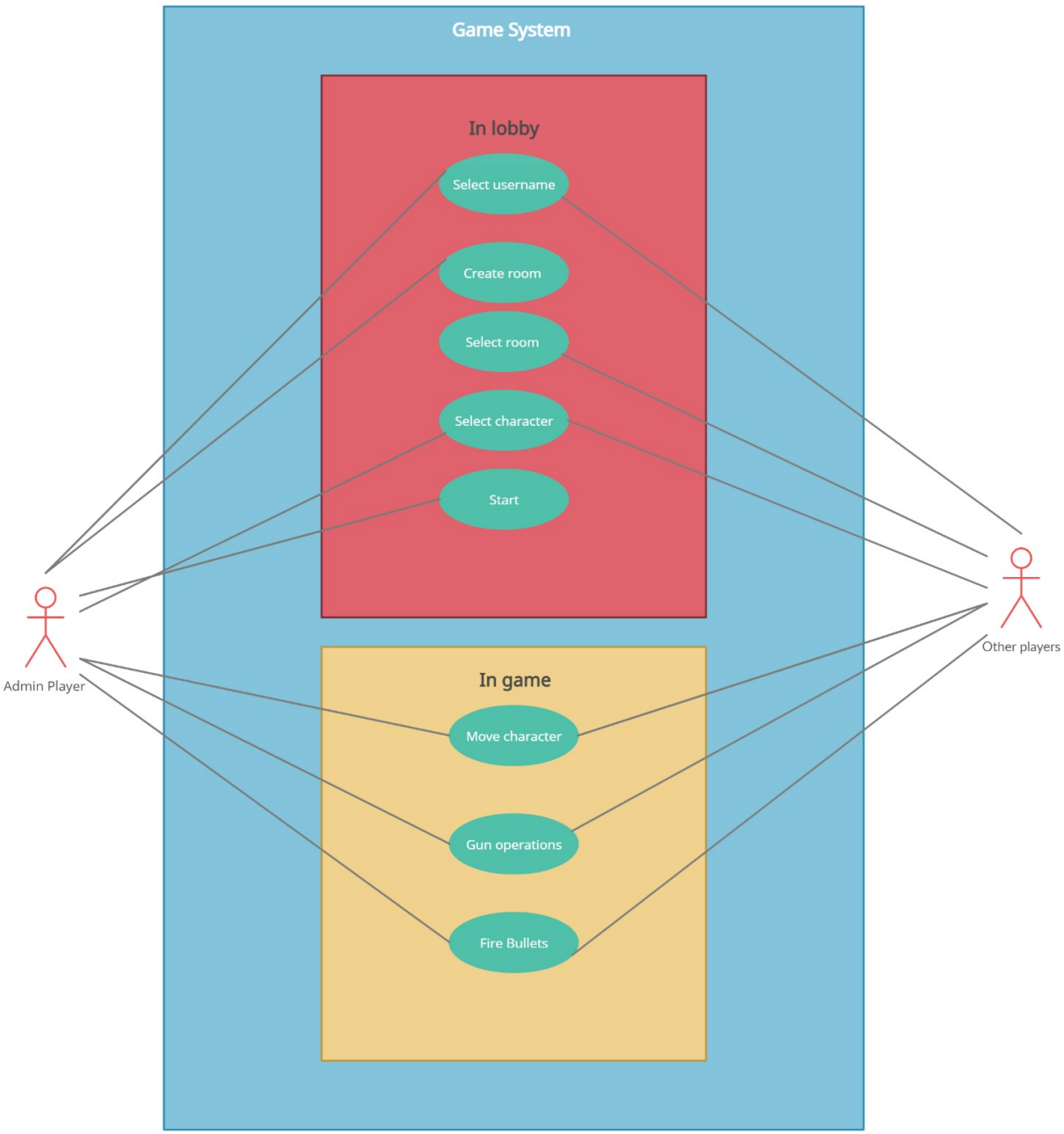
MELİH SAPMAZ

DIAGRAMS:

CONTEXT MODEL:



USE-CASE DIAGRAMS:



USE CASE TABLES

System	Game System
Use Case	Select Username
Actors	Admin Player,Other players
Data	Username of the players are stored in the game system and are hidden to appear in-game.
Stimulus	Players communicate with the system and request their usernames to be stored
Response	Usernames are stored in the system
Comments	Usernames must be more than 1 letter

System	Game System
Use Case	Create room
Actors	Admin player
Data	The admin player determines a room name and this name is stored in the system.Players who want to connect to the room are connected by seeing this name
Stimulus	Room creation section connects to unity server and creates the room that allows other players to connect
Response	Player data is sent to unity server
Comments	Room name must be greater than 1 letter

System	Game System
Use Case	Select Room
Actors	Other players
Data	Unity server sends room information to other players. The data of new players joining the room are kept in the system
Stimulus	Other players establish a communication link with the unity server and request their information to be forwarded
Response	Player information is sent to unity server
Comments	In this section, users can see each other's names

System	Game System
Use Case	Select Character
Actors	Admin Player,Other Players
Data	The data of the characters selected by the players are stored in the database
Stimulus	The character selection system communicates with the unity server.Request the transmission of character data
Response	The characters chose by the players are seen by other players
Comments	5 character can be selected

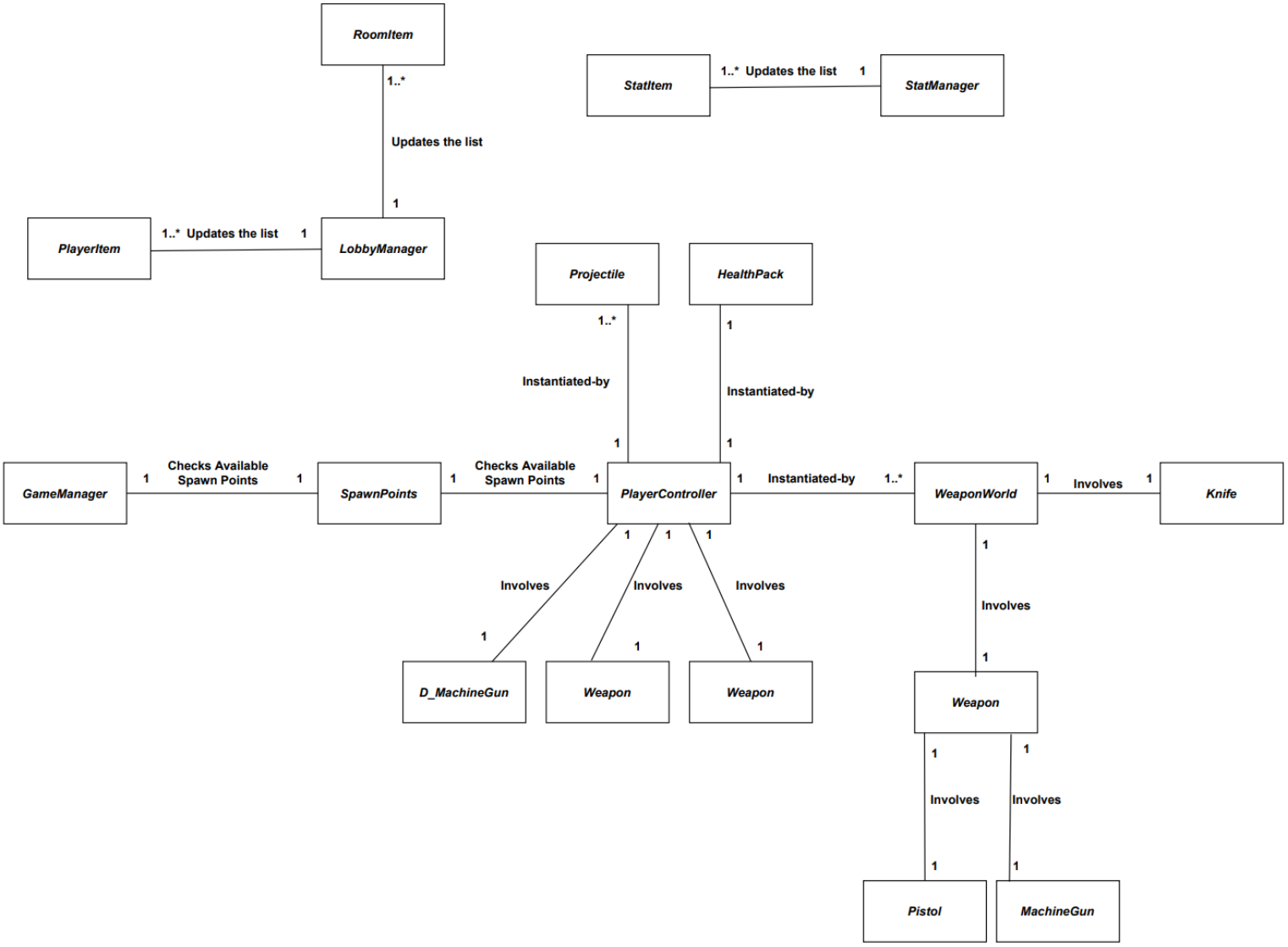
System	Game System
Use Case	Start
Actors	Admin Player
Data	Player information is sent to the game screen eg player name
Stimulus	By communicating with the unity server, the movements of the players are visible to each other
Response	The game begins
Comments	5 people can enter the game

System	Game System
Use Case	Move Character
Actors	Admin Player,Other players
Data	Player position information and movement instructions are stored in game system
Stimulus	Player position information is sent to the unity server so that other players can see the position changes
Response	The player moves
Comments	Player movement is done with the (W,A,S,D) keys on the computer. On the phone , there is a joystick

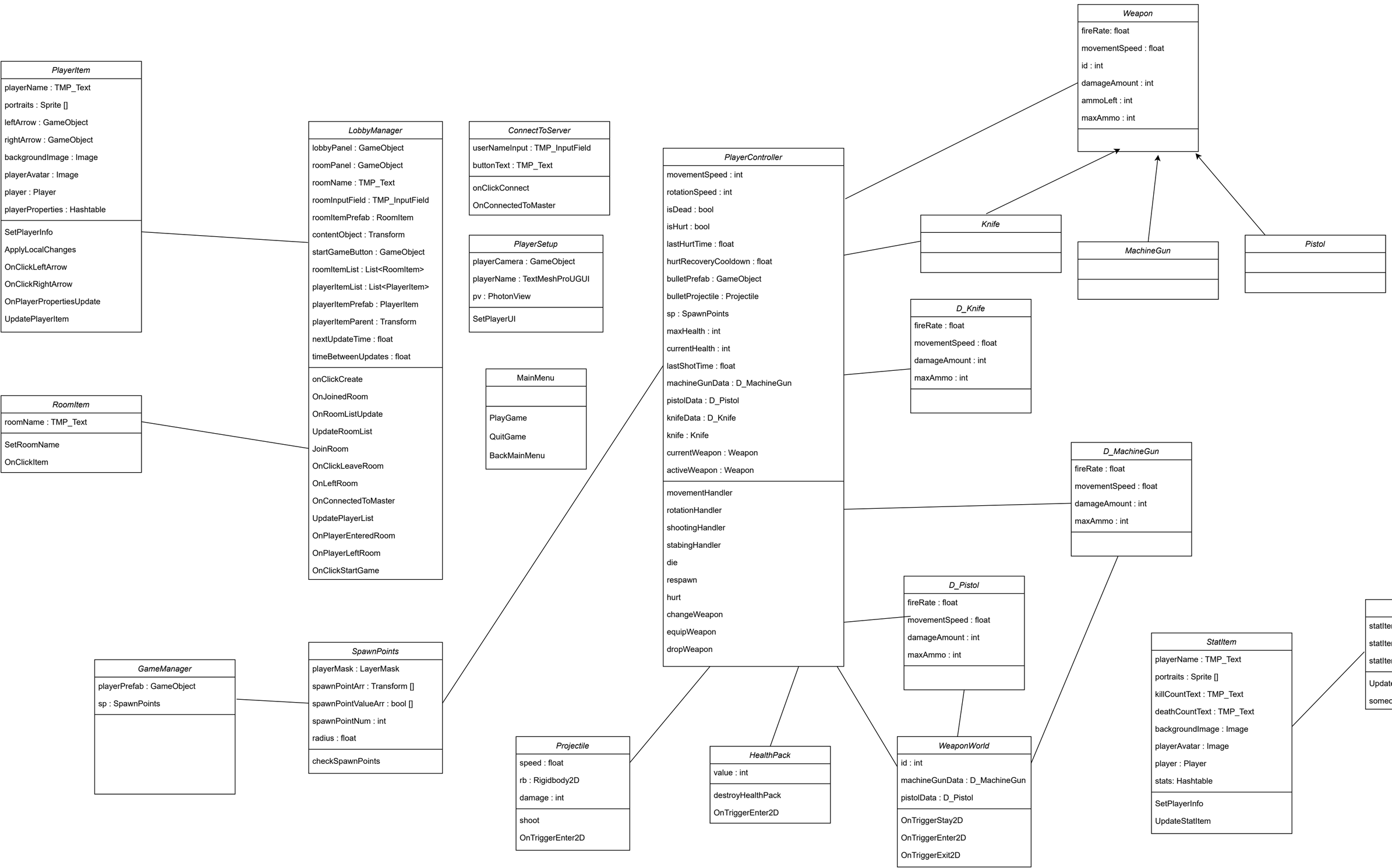
System	Game System
Use Case	Gun operations
Actors	Admin Player,Other players
Data	There are different types of weapons, these weapons have different characteristics ,the data is stored in the system.
Stimulus	Behaviors of weapons ,shooting etc. Interacts with unity server.When interacted,it goes to the opposite side as bullets
Response	The game begins
Comments	Pick up the gun throw the gun down shoot the bullet

System	Game System
Use Case	Fire Bullets
Actors	Admin Player,Other player
Data	Each weapons has a different feature,in line with these data , a certain amount of health is reduced in the character hit by the bullet, this data may change in the later parts of the game. Data is kept dynamically in the game system
Stimulus	The health information generated by the bullets hitting the characters communicates with the unity server and the characters whose health is depleted are eliminated
Response	Decreased health of characters
Comments	5 people can enter the game

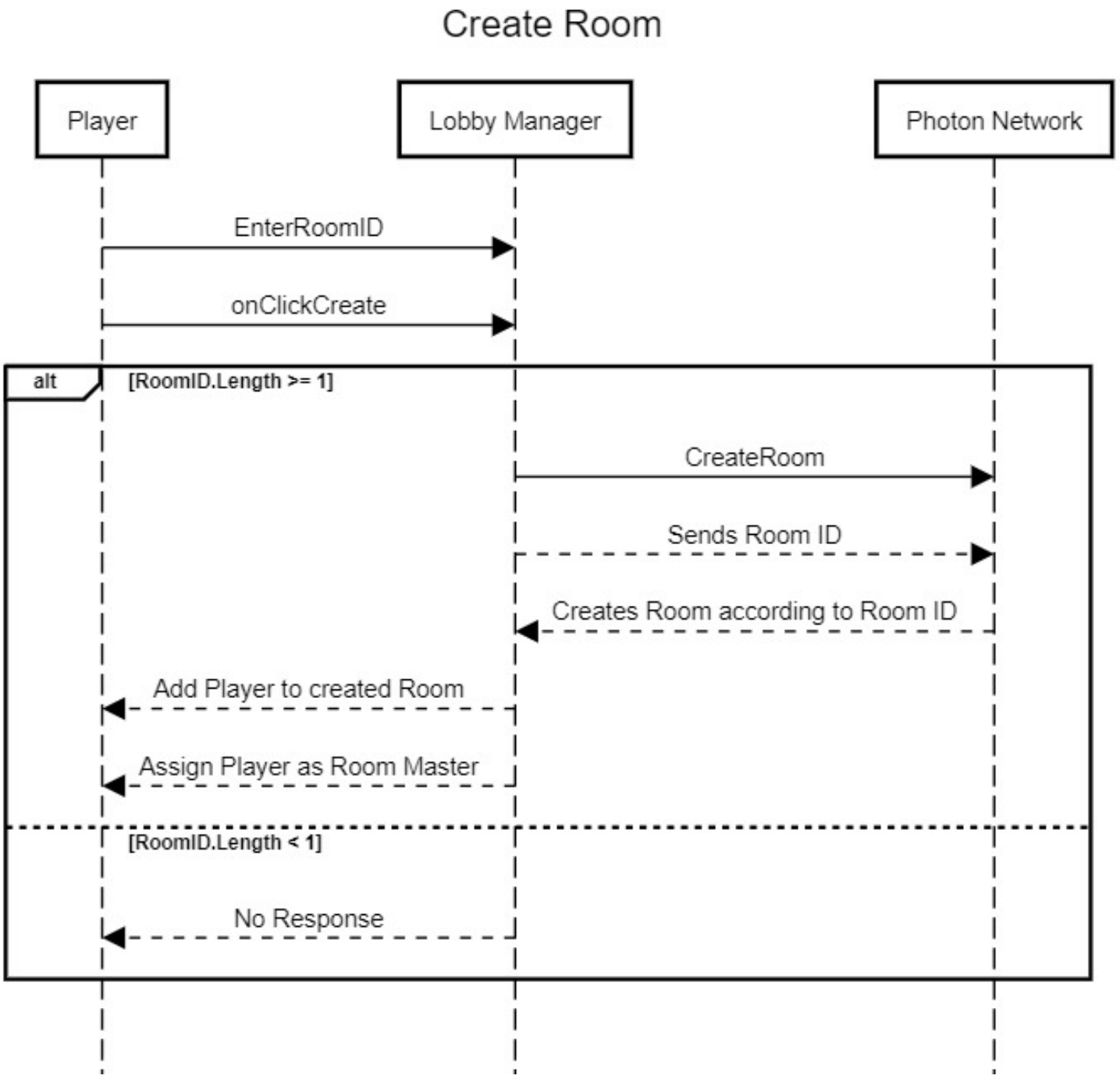
CLASS DIAGRAMS:



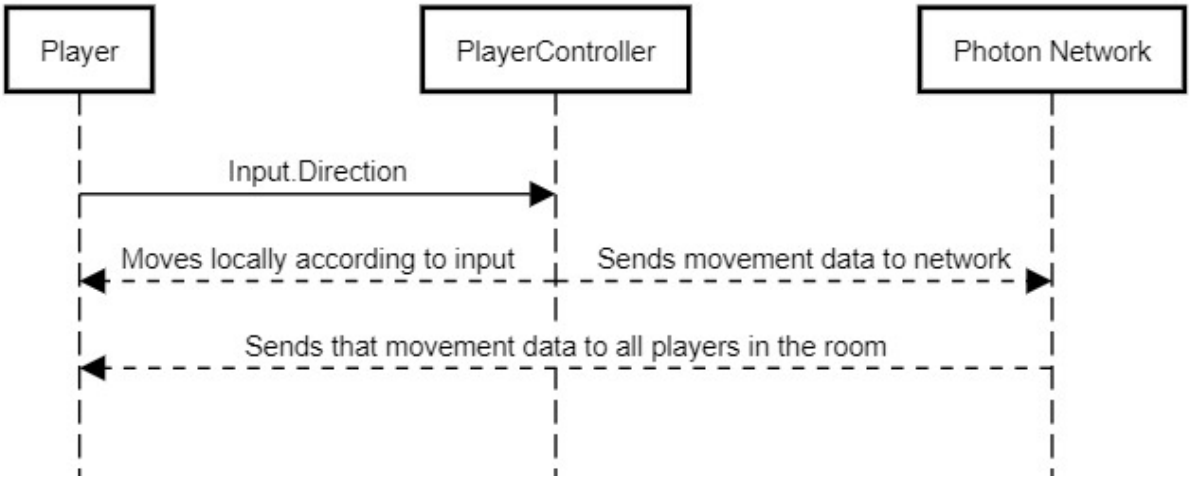
CLASS DIAGRAMS



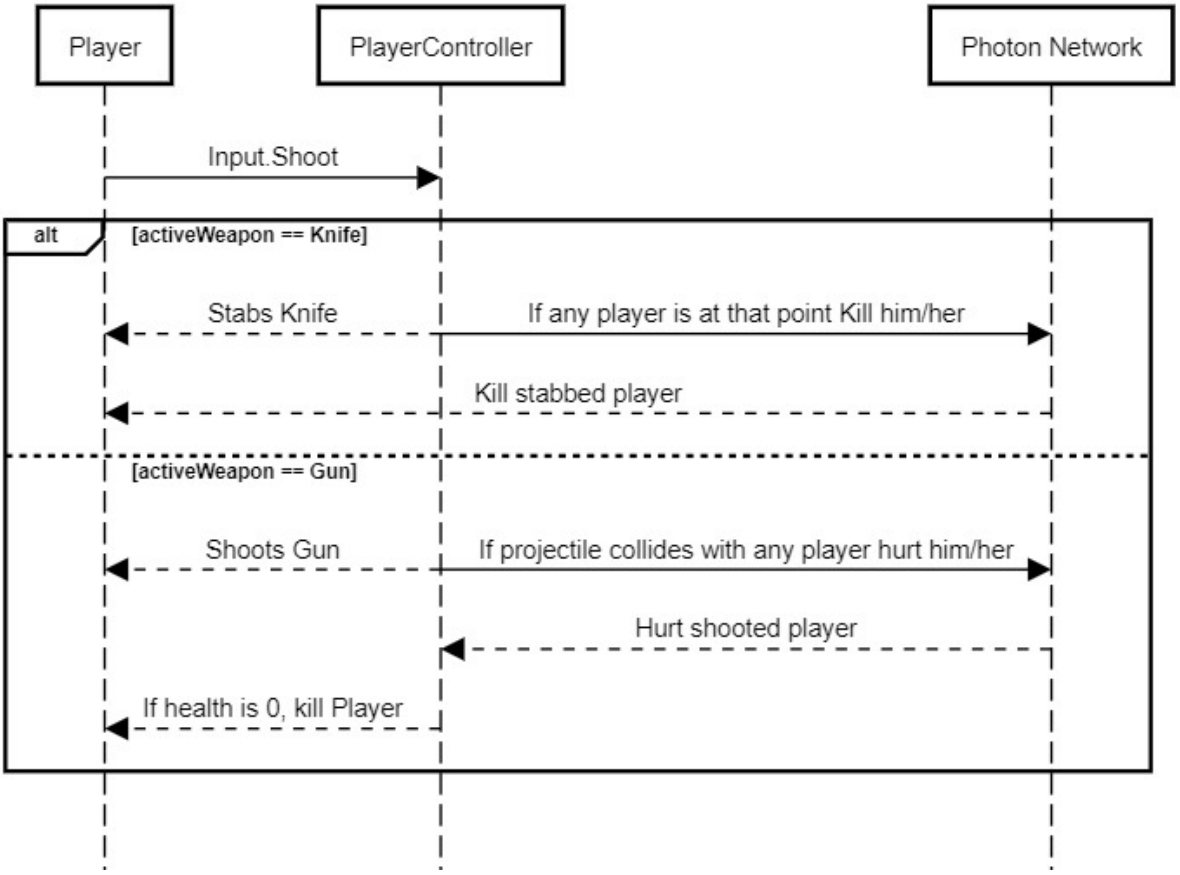
SEQUENCE DIAGRAMS:

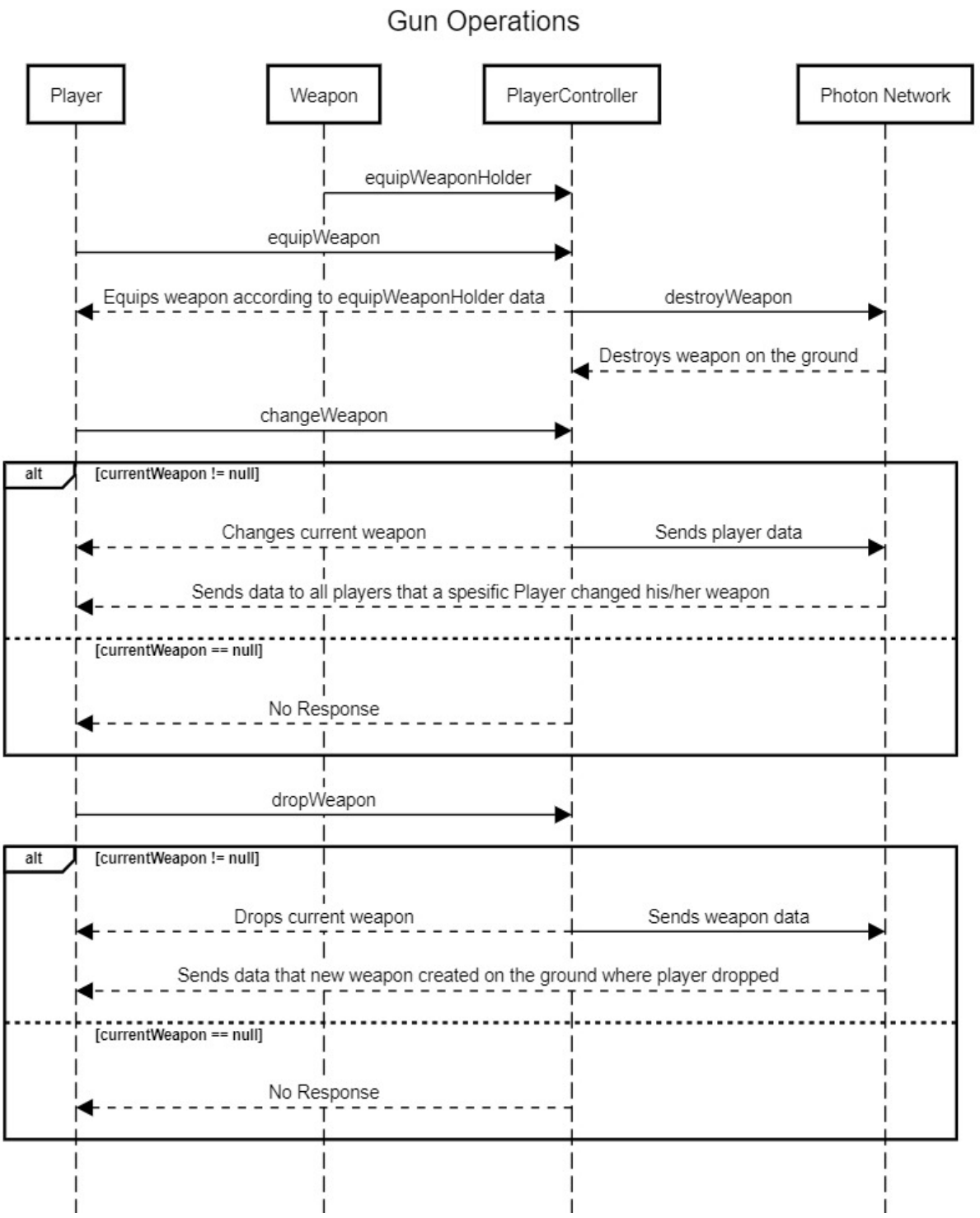


Move Character

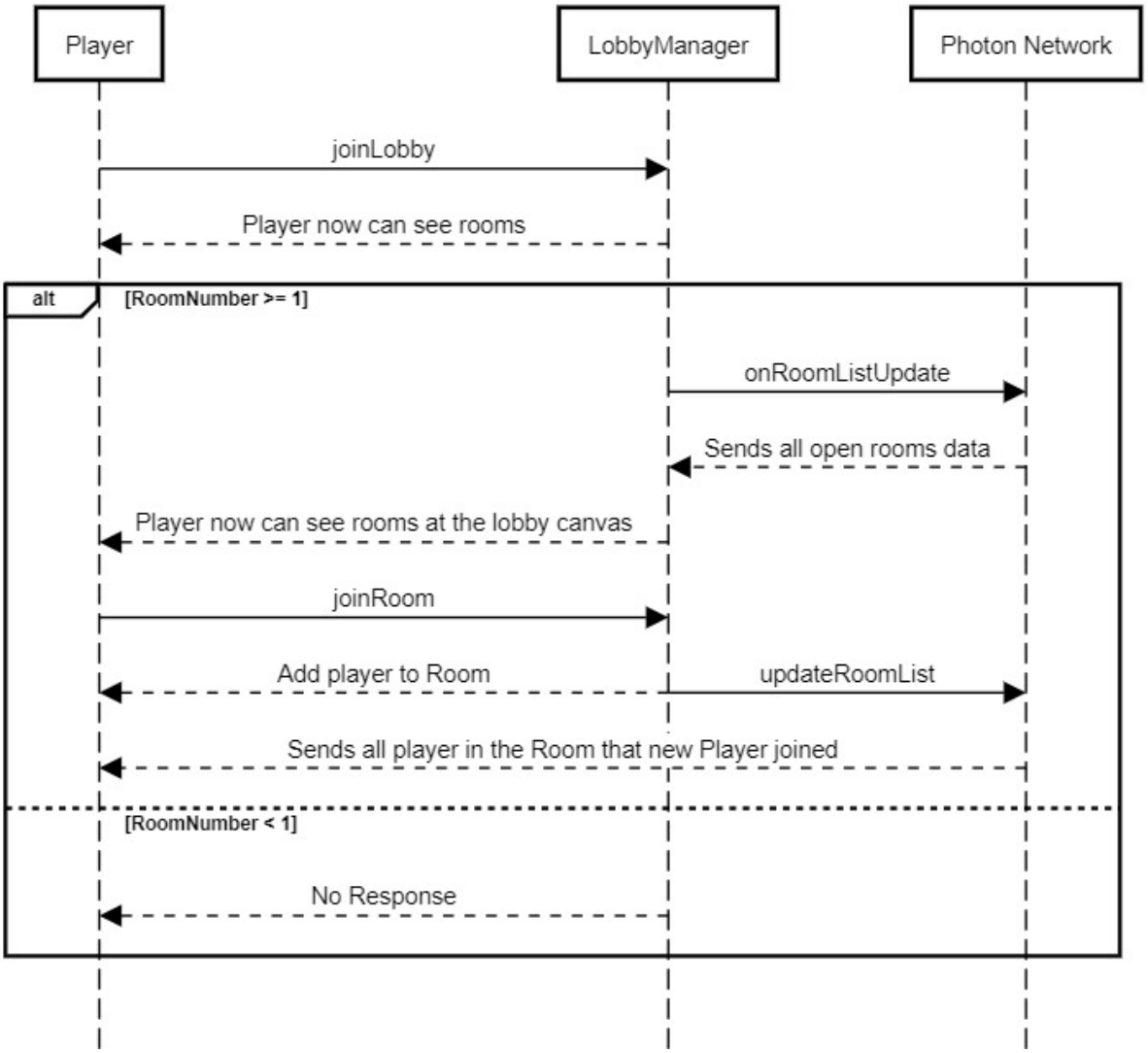


Fire Bullets

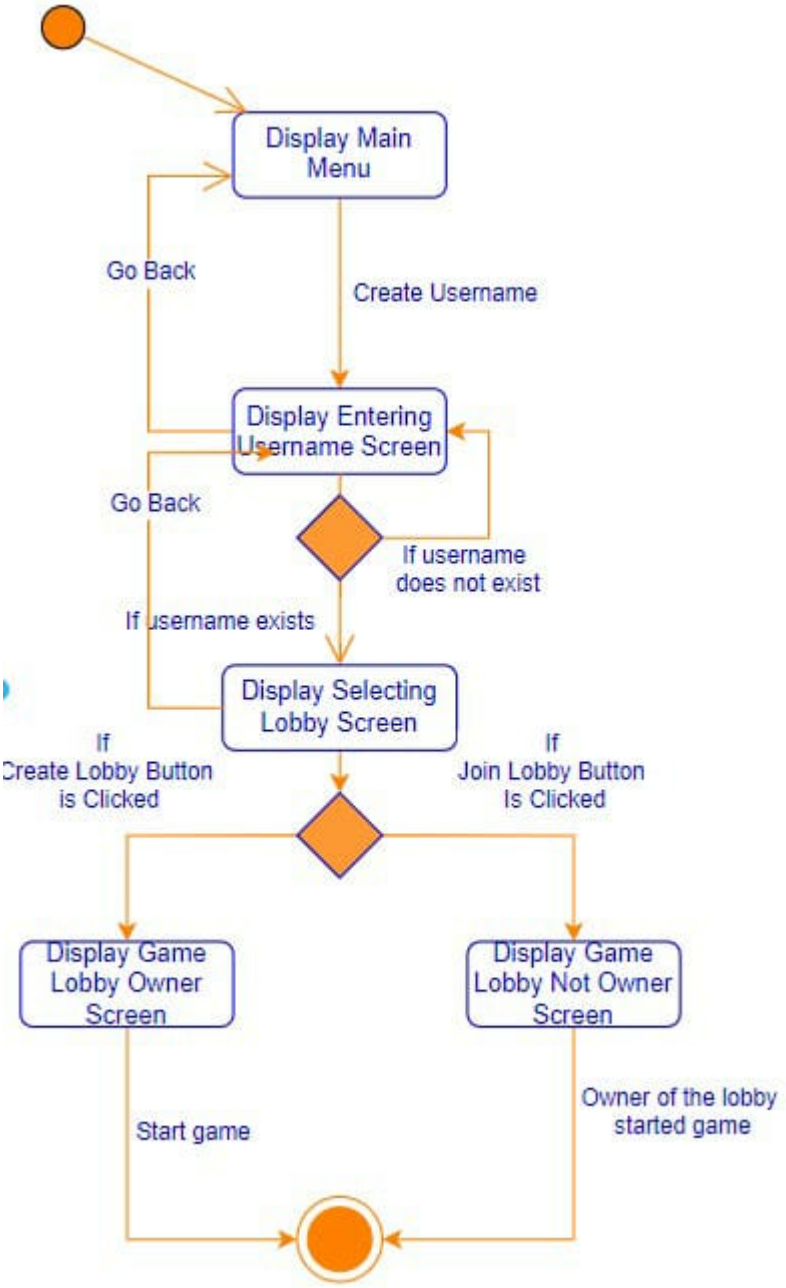



Gun Operations
Player
Weapon
PlayerController
Photon Network
equipWeaponHolder
equipWeapon
Equips weapon according to equipWeaponHolder data
destroyWeapon
Destroys weapon on the ground
changeWeapon
alt
[currentWeapon != null]
Changes current weapon
Sends player data
Sends data to all players that a spesific Player changed his/her weapon
[currentWeapon == null]
No Response
dropWeapon
alt
[currentWeapon != null]
Drops current weapon
Sends weapon data
Sends data that new weapon created on the ground where player dropped
[currentWeapon == null]
No Response

Select Room


Player
LobbyManager
Photon Network
joinLobby
Player now can see rooms
alt
[RoomNumber >= 1]
onRoomListUpdate
Sends all open rooms data
Player now can see rooms at the lobby canvas
joinRoom
Add player to Room
updateRoomList
Sends all player in the Room that new Player joined
[RoomNumber < 1]
No Response

STATE DIAGRAMS:



System Architecture Document :

We selected listener model for our System Architecture Document. Below, we explained each sub-system's advantages/disadvantages and which subsystem it communicates. We added figures in the end.

Lobby Management:

Lobby Management component uses Photon's Servers. Photon is a third party component for Unity Lobby Management component allows user to login to the network with the nickname he/she entered, create a room, join a specific room, join a random room, view room status of the each room in the room list, leave from the joined room, select a avatar in the room. It also hides the full rooms, sets the capacity for the each room (for us it is 5). Its disadvantage is, it only allows maximum 20 users to become online at the same time.

Lobby Management component communicates with Synchronization System, Game Stats Management and Spawn Management. Whenever a user is entered a room it communicates with the Synchronization System and sets the scene to room's synched scene. The avatar selected in the room is used in the Game Stats Management. Each players avatar is displayed there. If game has already started, it gives signal to the Spawn Management that a player is joined the game and it needs to be instantiated.

Spawn Management:

Spawn Management component controls weapon and player instantiations. Weapons have to be spawned on the map periodically. Also, players can drop weapons to the ground too. It spawns player prefab when a player is dead or it is joined to the game . It checks an available position for player to be spawned whenever it is necessary. We modeled this way. Because, game has to be dynamic and user has to be able to interact with the weapons. Otherwise, game would be so simple and boring since there is no dynamic elements inside it.

Spawn Management communicates with the Lobby Management, Synchronization System, Weapon Management.

It receives the signal from the Lobby Management and spawns the newcomer to an available position. It communicates with the Synchronization System whenever a new spawn action is executed. It receives the weapon data from the Weapon Management component and spawns the weapon according to that data.

Sound Settings:

Sound Settings benefits our game by allowing user to mute the complete sound or the music of the game. It also, plays the music at the menus.

It communicates with the Weapon Management and plays the current weapon's sound effects. It advantages our game by creating ambience for the user. We modeled Sound Settings component this way because we wanted user to feel realistic.

Weapon Management:

Weapon Management component holds data of the each weapon. This data contains weapon's ammo, damage amount, fire rate, movement speed. This component is essential for our game. Since after player shoots bullets, weapons ammo amount has to decrease. Otherwise game would be so unrealistic.

Weapon Management component communicates with the Sound Settings, Spawn Management, Projectile Management. It gives the current weapon's sound data to the Sound Settings. It gives the required weapon's data to the Spawn Management and Spawn Management instantiates this weapon at the required position in-game. It gives the current weapon's damage amount to the required bullet.

Synchronization System:

Synchronization System is the core component of our game. Without this component most of the other components in the system wouldn't work correctly. It is necessary, because every user needs to experience the same the event at the same time. We modeled this way. Because, our task is developing a multiplayer game. Without synchronization of the objects in the game, it wouldn't be possible. It uses Photon Views in order to synch the objects.

It communicates with the Spawn Management and Lobby Management. Whenever a new player joins to the room Synchronization System sets the scene to room's synched scene. With that approach each user views the same scene. It receives the Photon View's from the Spawn Management and synchs the game according to that data.

Game Stats Management:

Holds players kill, death and avatar type data by communicating with Player Health System for if player dies add death data to that player and kill data to other player, Lobby Management for taking avatar type of the player.

Player Health System:

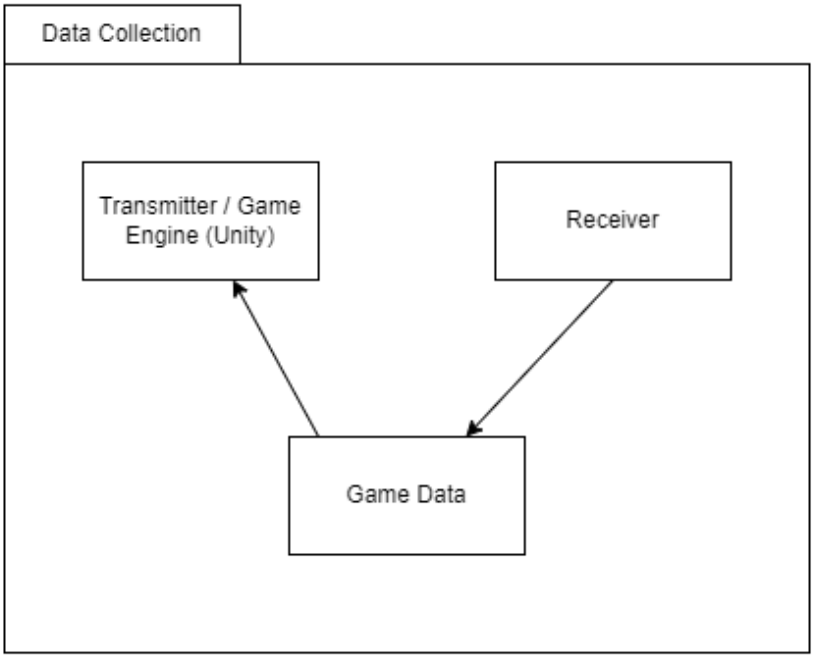
Holds data of players health for sharing because death is all that matters in this game, all systems works with this data. Communicates with Collision Management for if a projectile hits a player and hurts this could mean death or if player collides with Health Pack so that we can add health to player, Spawn Management for if player dies it shares that data with Spawn Management to resurrect player again at random Spawn Point.

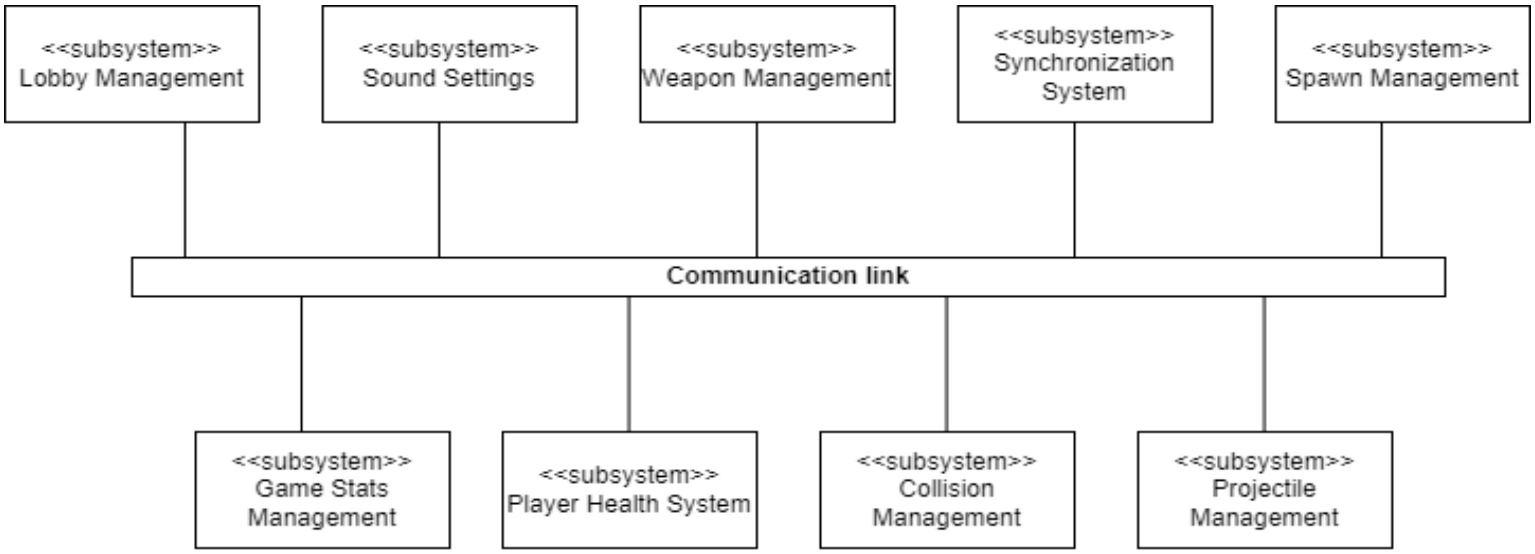
Collision Management:

Checks at every frame for if anything collides with anything so we can act according to that data. Communicates with Projectile Management if Player or stuff collides with projectile shares that data to destroy that projectile, Player Health System for if projectile/Health Pack collides with player act according to that, Synchronization System for synchronize every movement because in case of collision Players or stuff effect each others movement.

Projectile Management:

Holds projectiles speed, damage and which player shot it data. Communicates with Collision Management for if it collides with anything act according to that. Communicates with Weapon Management. Receives weapon's damage amount from it.



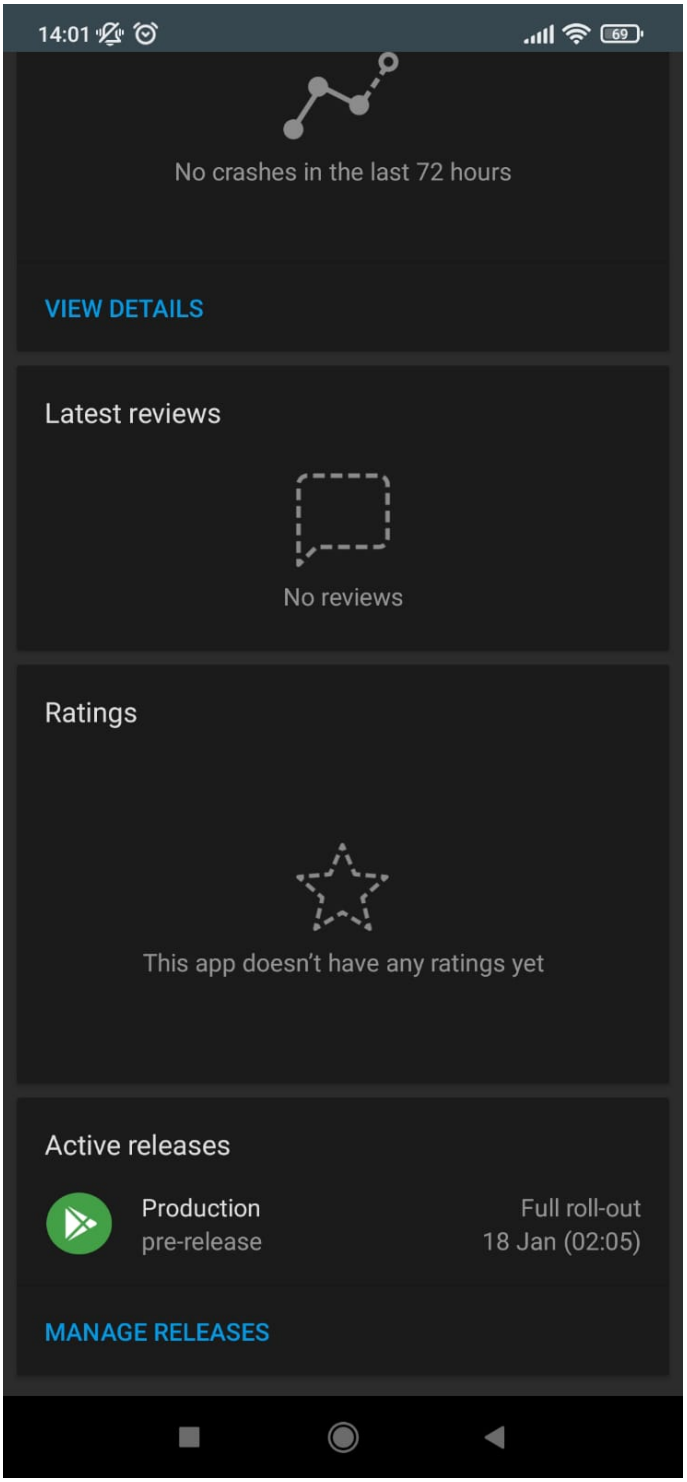

<<subsystem>>
Lobby Management
<<subsystem>>
Sound Settings
<<subsystem>>
Weapon Management
<<subsystem>>
Synchronization System
<<subsystem>>
Spawn Management
Communication link
<<subsystem>>
Game Stats Management
<<subsystem>>
Player Health System
<<subsystem>>
Collision Management
<<subsystem>>
Projectile Management

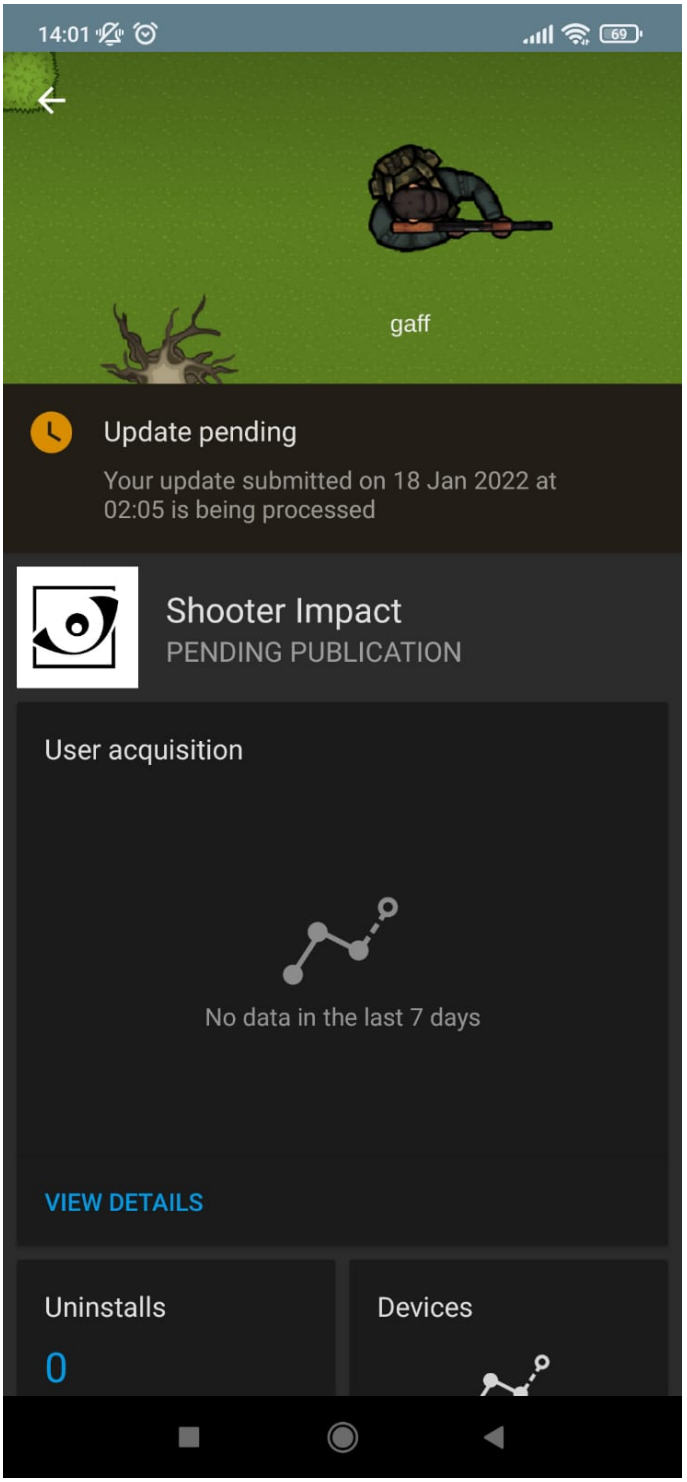
TEST CASES

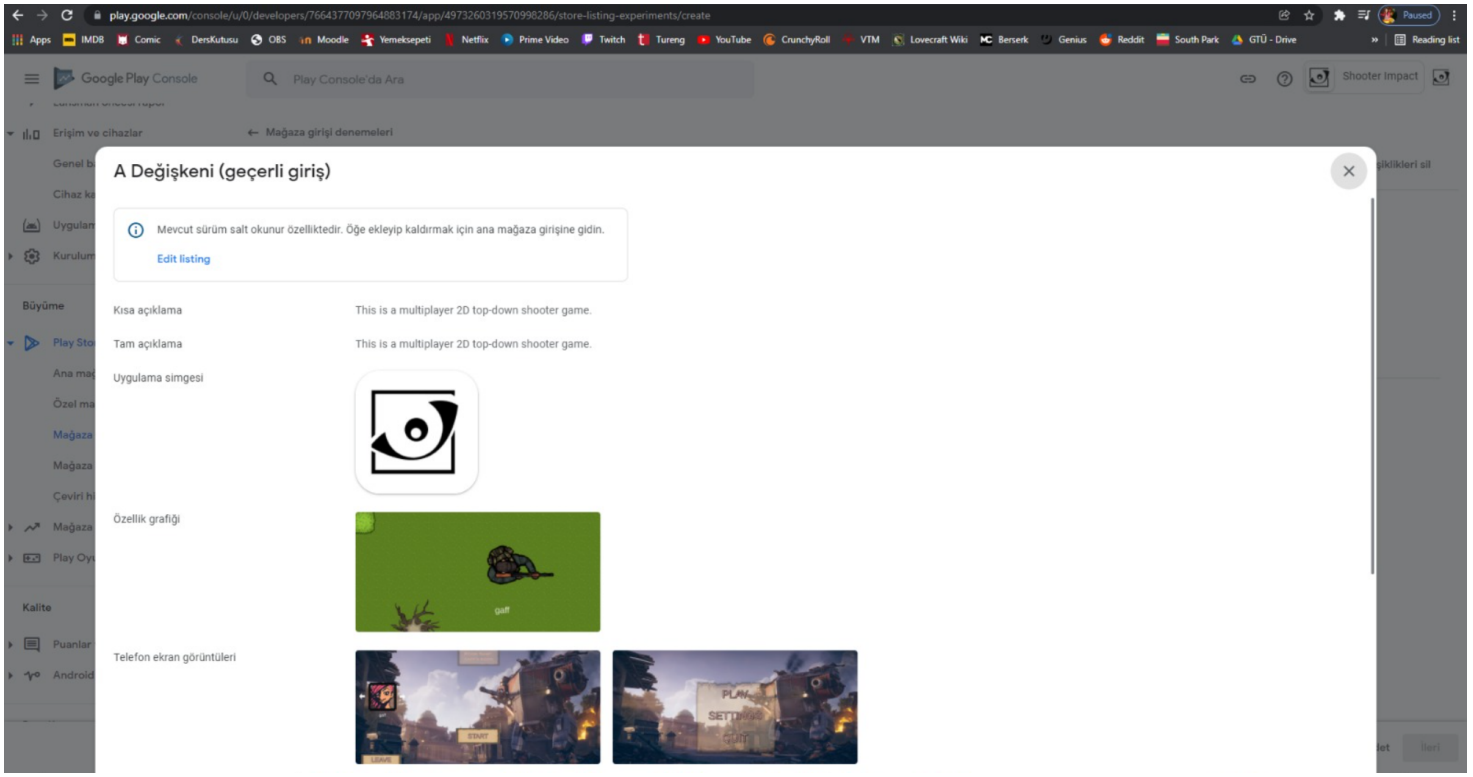
Test Case ID	Test Case Description	Expected Result	Pass/Fail
Lobby 1	Play button	Change scene to Server menu.	Pass
Lobby 1.1	Settings button	Change scene to Settings Menu.	Pass
Lobby 1.2	Quit button	Terminate the game.	Pass
Lobby 2	Nickname box	Take an input for a nickname.	Pass
Lobby 2.1	Connect button	Check if there's any input and connect to the server list.	Pass
Lobby 2.2	Back button	Change scene to Main Menu.	Pass
Lobby 3	Room name box	Take an input for a room name.	Pass
Lobby 3.1	Random Room button	Check if there's any room available and connect to the server.	Pass
Lobby 3.2	Create Room button	Create a room with given input.	Pass
Lobby 3.3	Back button	Change to nickname scene.	Pass
Lobby 3.4	Main Menu Button	Change scene to Main Menu.	Pass
Lobby 3.5	Server List	Connect the user to the selected server.	Pass
Lobby 4	Sliding through avatars	Select an avatar.	Pass
Lobby 4.1	Start button	Start the game.	Pass
Lobby 4.2	Leave button	Disconnect user from server and connect back to server list.	Pass
Game 1	Left Joystick	Character movements.	Pass
Game 1.1	Right button	Pick up a gun.	Pass
Game 1.2	Right Joystick	Shoot	Pass
Game 1.3	Middle button	Drop a gun.	Pass
Game 1.4	Main Menu	Leave the server. Change scene to Main Menu.	Pass
Game 1.5	Left button	Change the gun.	Pass
Game 1.6	Scoreboard button	Shows the scoreboard	Pass

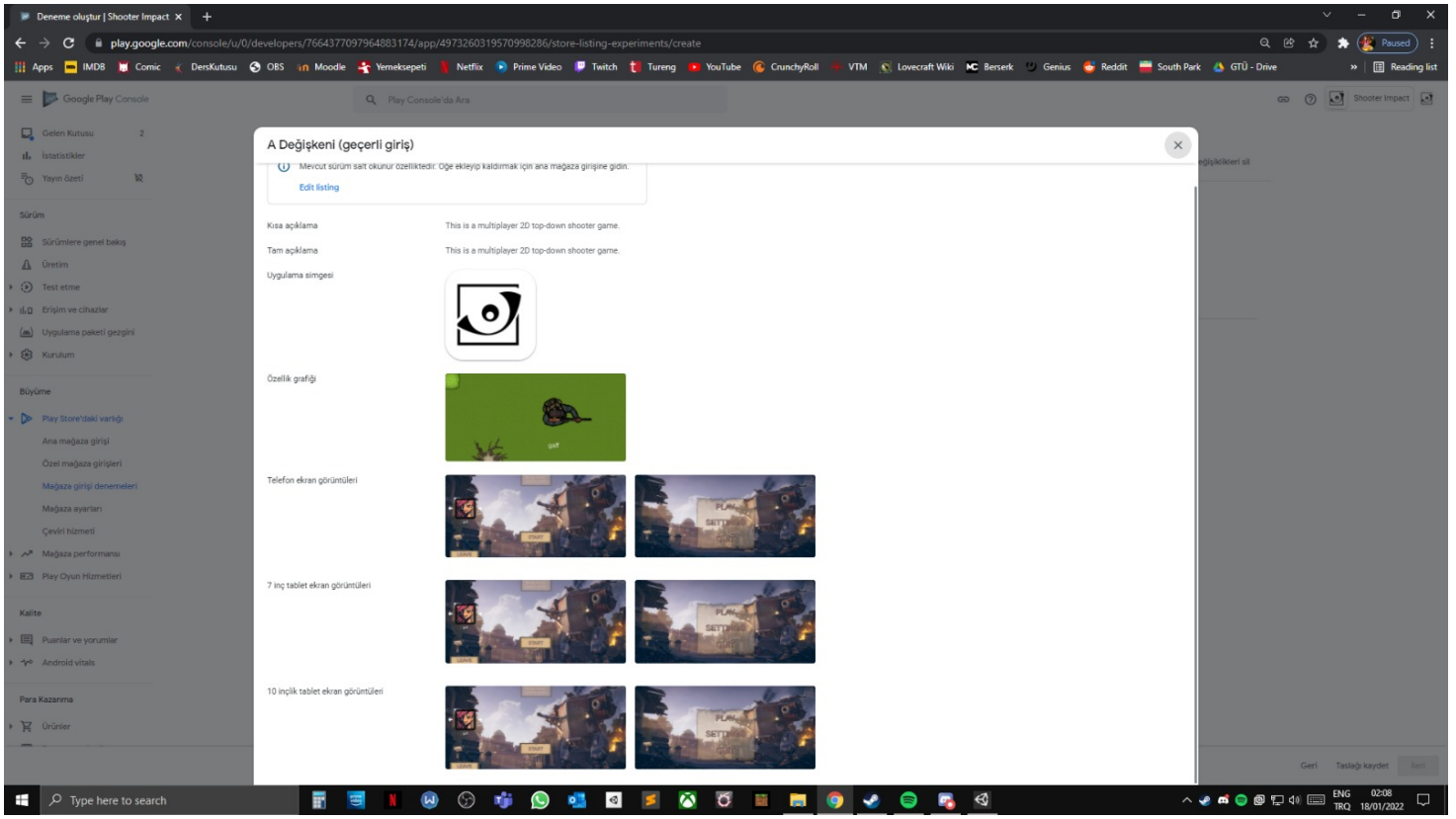
Is your project accessible to potential users?

Screenshots of the game in google play






play.google.com/console/u/0/developers/7664377097964883174/app/4973260319570998286/store-listing-experiments/create
Google Play Console
Play Console'da Ara
Shooter Impact
Mağaza girişi denemeleri
A Değişkeni (geçerli giriş)
Mevcut sürüm salt okunur özelliktedir. Öğe ekleyip kaldırmak için ana mağaza girişine gidin.
Edit listing
Kısa açıklama
This is a multiplayer 2D top-down shooter game.
Tam açıklama
This is a multiplayer 2D top-down shooter game.
Uygulama simgesi
Özellik grafiği
Telefon ekran görüntüleri


Deneme oluştur | Shooter Impact
play.google.com/console/u/0/developers/7664377097964883174/app/4973260319570998286/store-listing-experiments/create
A Değişkeni (geçerli giriş)
Edit listing
Kısa açıklama
This is a multiplayer 2D top-down shooter game.
Tam açıklama
This is a multiplayer 2D top-down shooter game.
Uygulama simgesi
Özellik grafiği
Telefon ekran görüntüleri
7 inç tablet ekran görüntüleri
10 inçlik tablet ekran görüntüleri
Type here to search