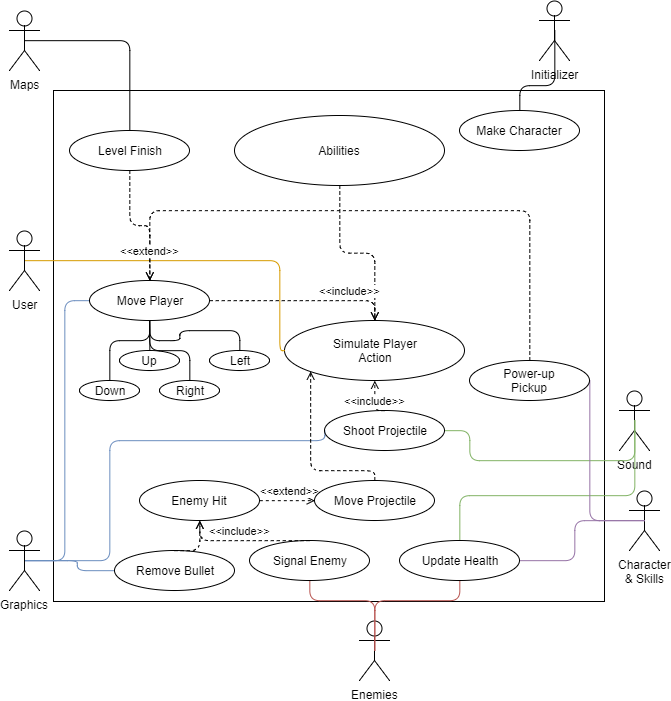
Baylus Tunnicliff Mark \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/50

## Player Management \_\_/3

My feature is managing the player specifically. I receive direct input from the user to stream line the process. I am responsible for moving the player, using the combat stats of the character to fire projectiles when the player gives the proper input, and managing these projectiles. Ensuring the player doesn’t move out of bounds, signaling when an enemy is hit by a projectile, and picking up power-ups are some of the outlier functions that I will be implementing.

## Use case diagram with scenario \_\_14

### Use Case Diagrams



### Scenarios

**[You will need a scenario for each use case]**

**Name:** Simulate Player Action

**Summary:** User’s input is taken to model the action of the character in the game.

**Actors:** User

**Preconditions:** Initializer has give init() call

**Basic sequence:**

Obtain user input

If input is:

Directions, move corresponding direction

Fire Key, shoot projectile

Exceptions

If key pressed is found in ability list,

Activate ability

**Post conditions:** Action is performed

**Priority: 1**

**ID:**

**Name:** Move Player

**Summary:** Moves the player around the map and signals interactions when colliding with certain cases.

**Actors:**

**Preconditions:** Simulate Player Action received movement input

**Basic sequence:**

If input signal was direction signal,

For down, move down

For up, move up,

For left, move left

For right, move right

**Exceptions:**

If player collides with x after move:

Wall, then move up to wall,

Level end, then exit level

Power-up, then pickup powerup

**Post conditions:** Player move or collisions checked.

**Priority: 1**

**ID:**

**Name:** Shoot Projectile

**Summary:** Player attempts to shoot using their weapon.

**Actors:** Sound, Graphics

**Preconditions:** Weapon Fire Triggered

**Basic sequence:**

Fire weapon in direction of mouse

Update graphics to render projectile in view,

Signal sound to trigger weapon fire.

**Exceptions:**

If player pacified or in I-frame,

Don’t fire

**Post conditions:** Bullet fired or exception met,

**Priority: 2**

**Name:** Move Projectile

**Summary:** Moves projectiles on a clock created upon init() call.

**Actors:**

**Preconditions:** Init() call, timer tick

**Basic sequence:**

If projectileList is not empty,

Move each projectile according to the stats of projectile,

**Exceptions:**

If projectile collides with non-player,

If projectile collides with enemy,

Signal Enemy to damage enemy object

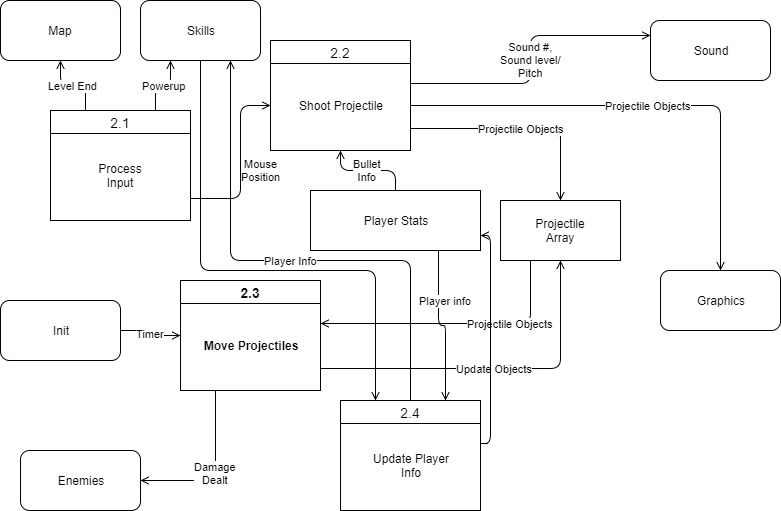
Remove projectile

**Post conditions:** Projectiles moved.

**Priority: 2**

## Data Flow diagram(s) from Level 0 to process description for your feature \_\_\_\_\_\_\_14

### Data Flow Diagrams



### Process Descriptions

Process Input:

WHILE Input is direction key AND movement in direction does not collide player

Move player direction \* moveSpeed

END WHILE

IF player collide with level end, or player collide with powerup,

If level end, signal map level end,

If powerup, signal skills powerup #,

END IF

IF input is fire key,

Shoot projectile using mouse location

ENDIF

IF ability lookup of input is true,

Execute ability,

update ability cooldown.

END IF

Shoot Projectiles:

Get char info,

Use bulletFireSpeed to calculate volume and pitch modifiers of sound,

Signal Sound modifiers for sound and pitch and sound key

Create bulletObject with relevant bullet info,

Add new bullet to bullet array,

Signal Graphics to update projectile array,

Move Projectiles:

FOR all projectiles in projectile array

Use bullet info to calculate bullet movement,

IF new bullet move collides with non-player, collide-type,

IF collide with enemy,

Signal Enemy Damage,

END IF

Remove bullet from projectile array,

END IF

END FOR

Signal Graphics to update models.

## Acceptance Tests \_\_\_\_\_\_\_\_9

**Example for fire projectile feature**

Run feature 1000 times sending output to a file. Each time run, test bulletSpread, bulletSpeed, bulletDmg, bulletSize, and bulletDrop for data proving consistency of bulletDeviation results.

The output file will have the following characteristics:

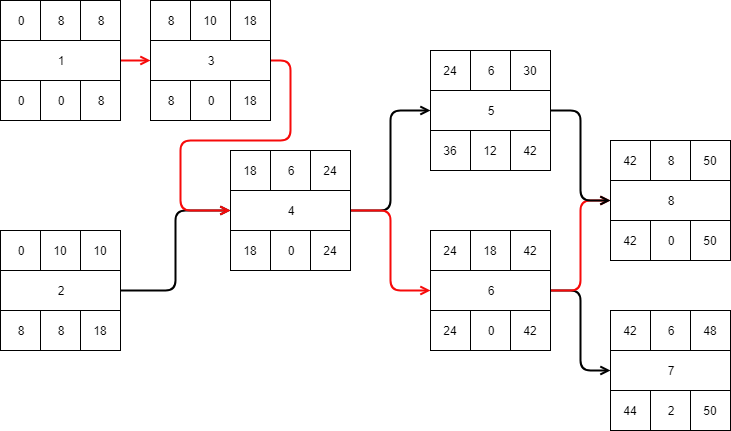
* Max number: 30
* Min number: -30
* 75% of numbers on averages should end up between -15 and 15, no more than 65% will end up either positive/negative.

## Timeline \_\_\_\_\_\_\_\_\_/10

### Work items

|  |  |  |
| --- | --- | --- |
| Task | Duration (PHrs) | Predecessor Task(s) |
| 1. Environment Study | 8 | - |
| 2. Collaborative Design | 10 | - |
| 3. Resource Design | 10 | 1 |
| 4. Ability Design | 6 | 2, 3 |
| 5. Character Design | 6 | 4 |
| 6. Programming | 18 | 4 |
| 7. Feature Redesign/Overhaul | 6 | 6 |
| 8. Testing | 8 | 5, 6 |

### Pert diagram



### Gantt timeline

