

# Game Design Document Combined

## Controls

Action	Gamepad (XBox Layout)	Keyboard & Mouse
Aiming	Right stick	Mouse
Movement	Left stick	Arrows / WASD
Shoot	Right Trigger	LMB
Ability	Right Shoulder	LShift
Interact	Button South	E
Pause	Start	Escape
Change arrow	Button North	R

## Genre and synopsis

### Genre

**2D Topdown Extraction Action** – a game focused on exploration, combat, and resource management from a top-down perspective. Players navigate dungeons, defeat enemies, collect loot, and extract rewards, often under risk of losing resources.

### Synopsis

The game is set in a **fantasy medieval world**. The protagonist travels through a portal into a magical dimension in the form of a dungeon inhabited by various hostile creatures. Players must explore rooms, overcome enemies, and gather resources while surviving the dangers of the dungeon.

## Player Progression



### Progression

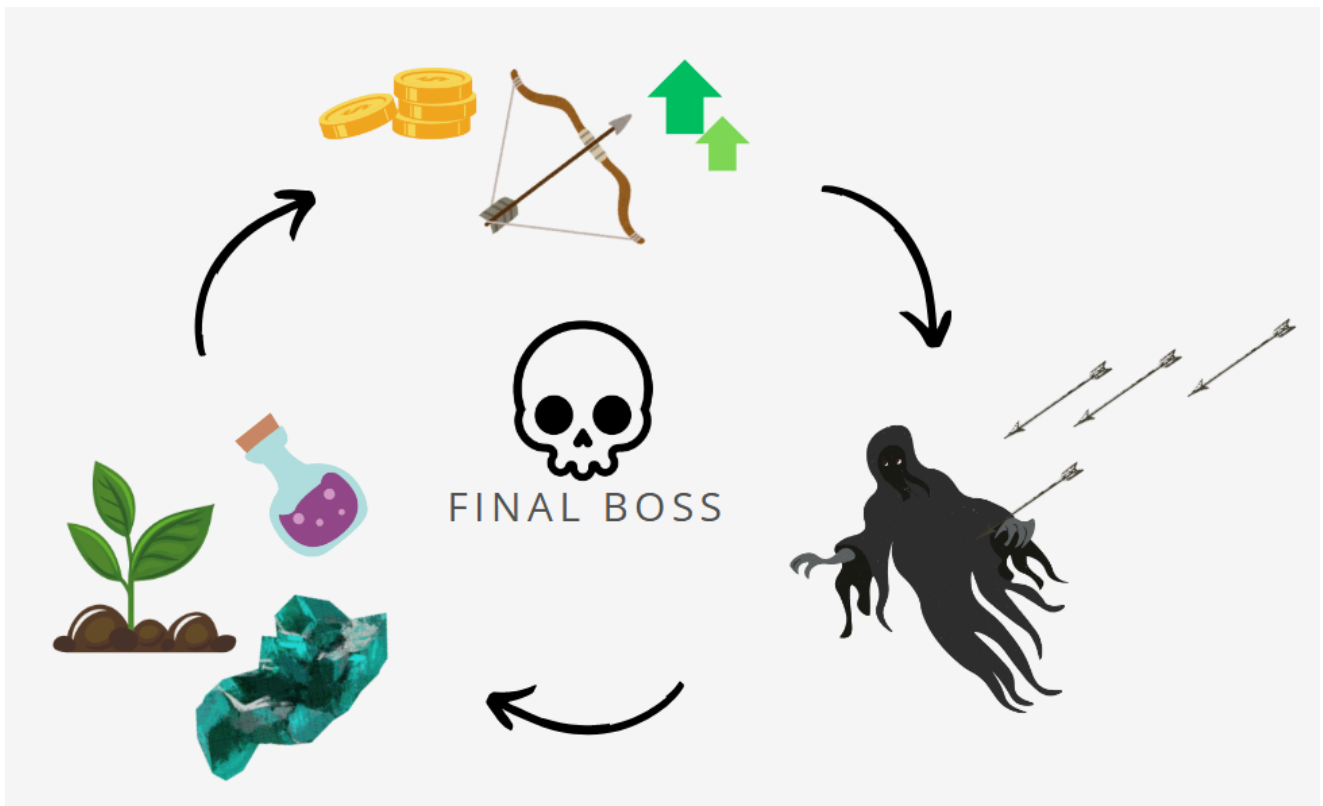
The progression system defines the sequence of challenges and key milestones the player experiences during a run, from the beginning to the end of the game.

## Sequence of a Run

1. **Entering the Dungeon** – The player begins the run at the dungeon entrance with starting resources, abilities, and equipment. Before each run, the player can also stock up in the shop with all necessary items.
2. **Room Exploration** – players explore rooms, encounter enemies, NPCs and find consumables and lore-related things.
3. **Combat Encounters** – players fight enemies using movement, attacks, and consumables to survive.
4. **Progression Milestones** – players reach key rooms. This can include unlocking new areas, gaining new abilities, or facing stronger enemies. Additionally, players can **exit the dungeon** to safely carry out collected resources without losing them, converting it to gold for getting better loadout next time.
5. **Boss Encounters** – major battles occur at designated points. Bosses test the player's skill in combat, positioning, and resource management.
6. **Run Completion** – The player exits the dungeon either successfully or fails by losing all health/resources. Rewards are granted based on achievements and collected items.

Each run is accompanied by skill progression and **meta-progression**, such as unlocking new loadouts with different weapon modifiers, which influence the player's strategy.

## Game Loop



## Style

---

## Visual References

ref-general-moodboard.png

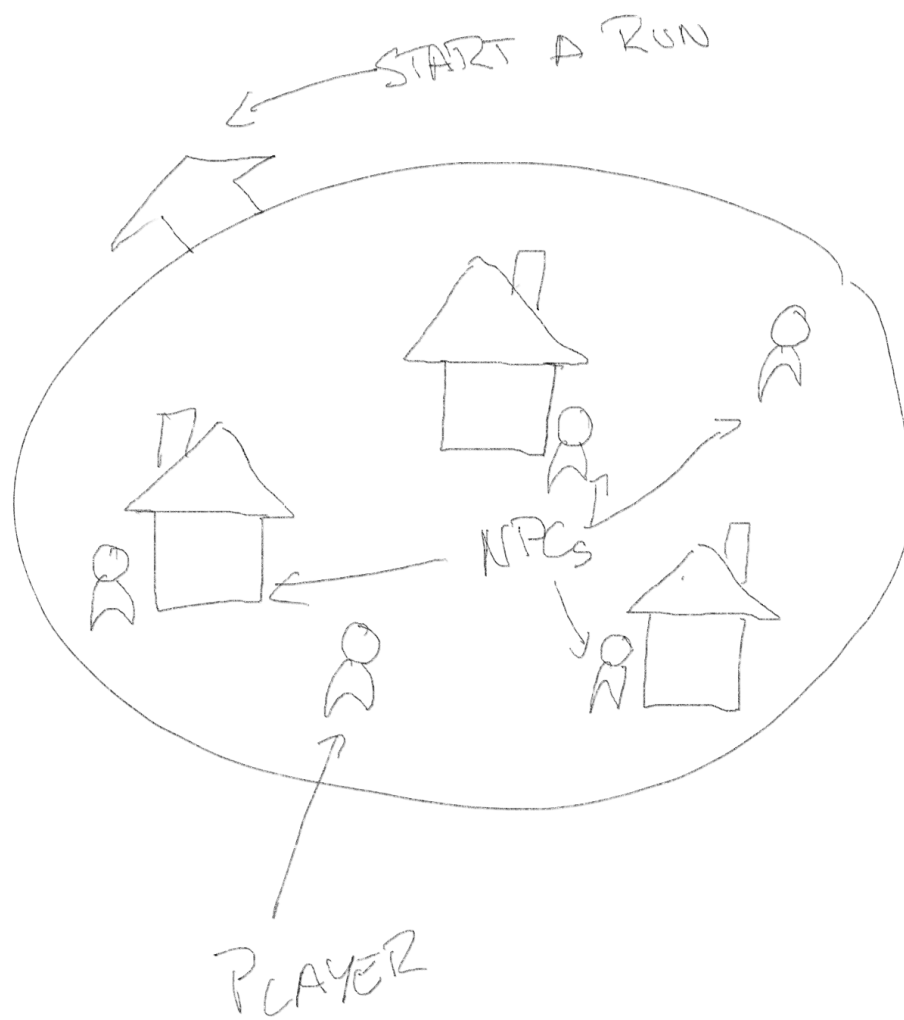
A sprite stacking reference:



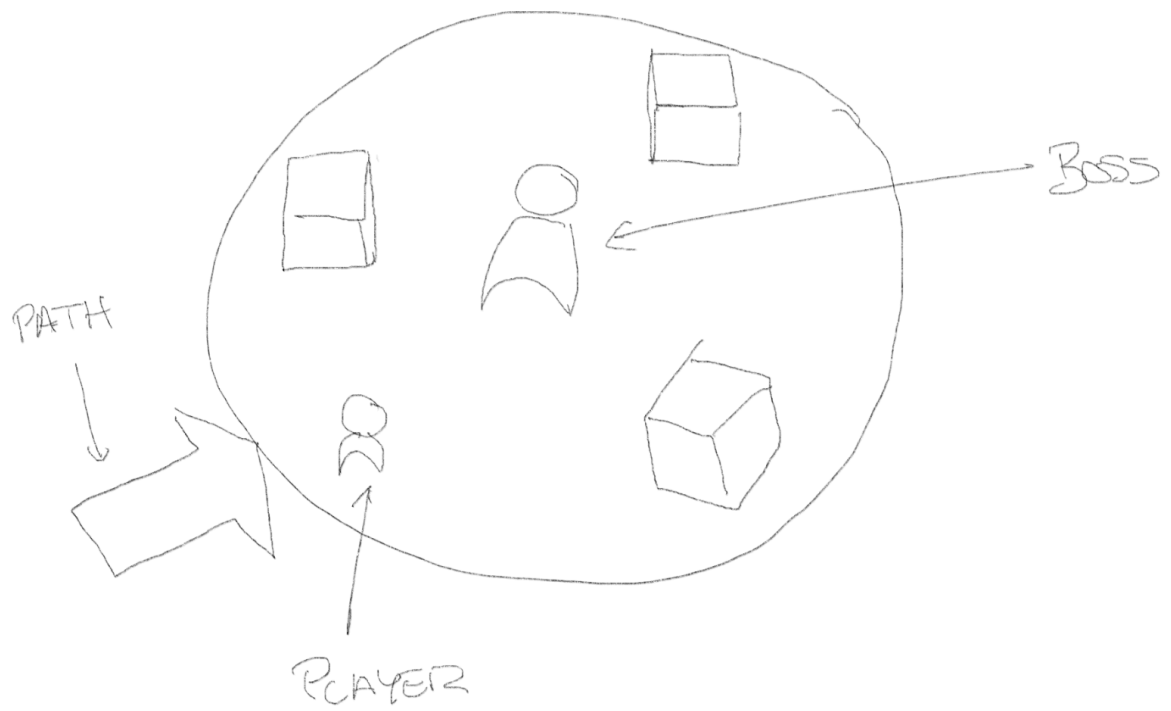
## In-game world

<In-game world description>

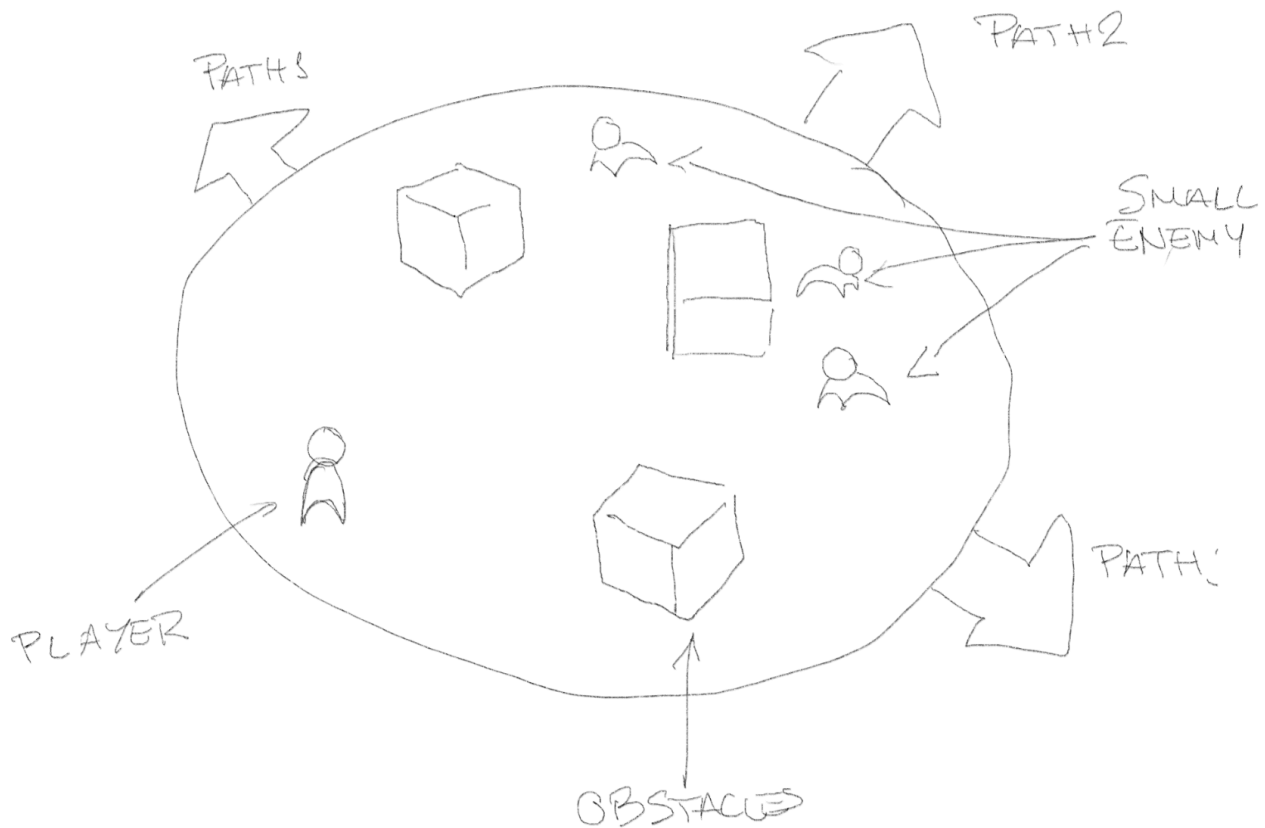
Hub sketch:



**Boss room sketch:**



**Small enemies room sketch:**



## Sprite Stacking

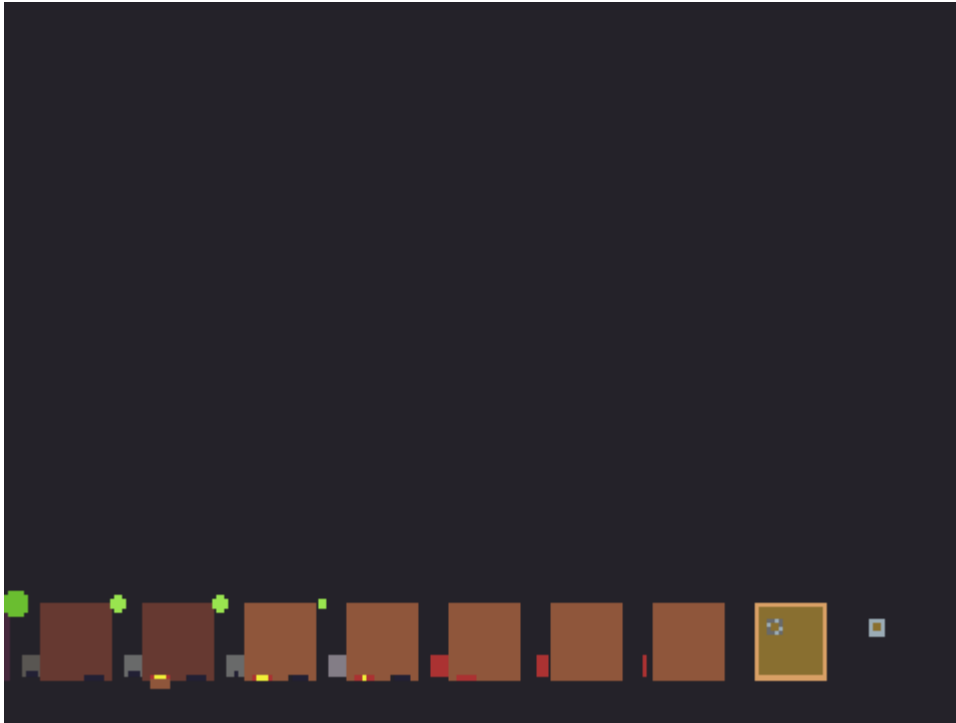
---

**Sprite Stacking** – is a rendering technique that simulates a three-dimensional look using two-dimensional sprites. It creates the illusion of depth by layering multiple 2D images (sprites) on top of each other with a small offset along the Z or Y axis.

### How it works

Each sprite layer represents a “slice” of an object (for example, a part of a tower, tree, or rock). During rendering, all these slices are positioned at equal intervals on top of each other. As the viewing angle or lighting changes, the stacked layers create the impression that

the object has real depth.



## Systems

### Boss Fight

**Boss Fight System** – a specialized system that manages encounters with powerful enemies, defining their behavior, attack patterns, and the conditions for victory or defeat. It builds on the core combat mechanics to create high-stakes, strategic challenges for the player.

### Description

#TODO Description of boss behaviour

### Related Systems

- [Combat](#)

### Combat

**Combat System** – a core system that encompasses all mechanics related to dealing damage to enemies and avoiding damage from them.

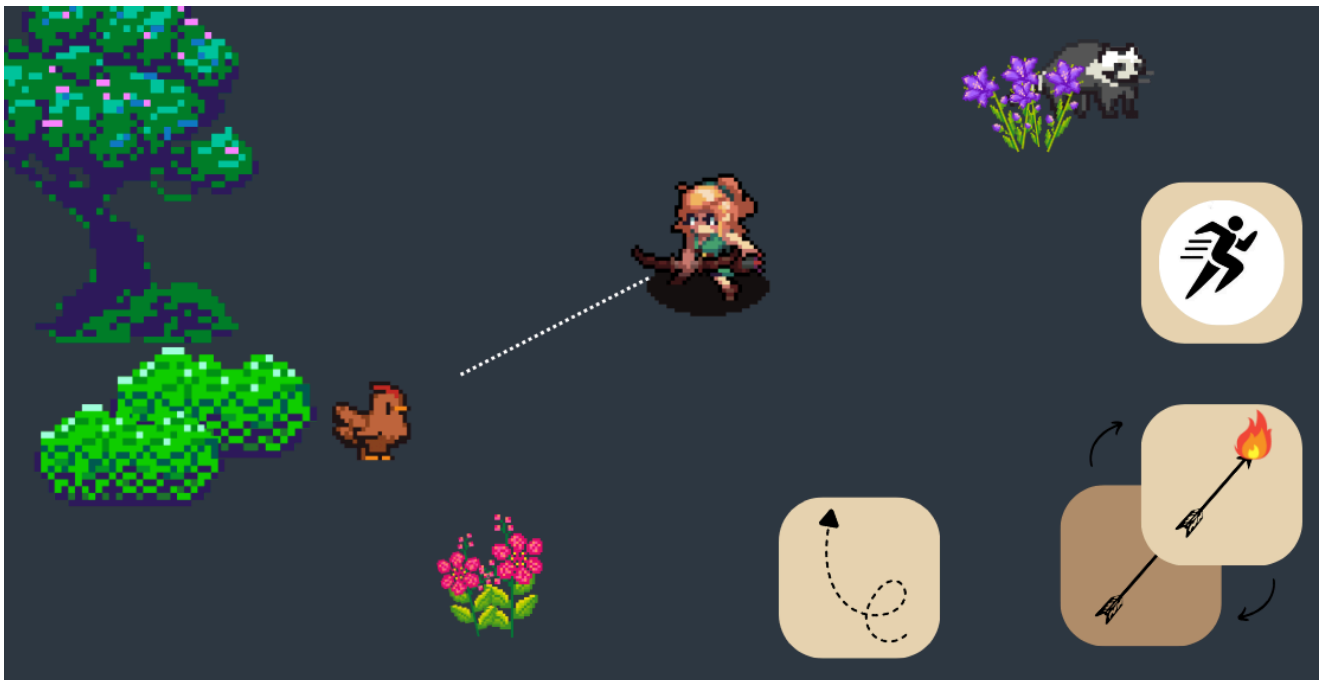
### Description



The Combat System governs how the player interacts with hostile entities in the magical realm. It includes aiming and shooting projectiles, managing consumable arrows, modifying arrow trajectories, and strategically selecting projectile types. Combat is entirely ranged, with no melee interactions, and emphasizes positioning, timing, and resource management.

Players must consider their movement, the type of arrows equipped, and active arrow path modifiers to overcome various enemy patterns. Stronger arrows or modifiers provide tactical advantages but are limited in supply, making planning and adaptability crucial. The system also incorporates movement-based survivability mechanics such as invulnerability frames from jumping or using abilities, allowing the player to avoid incoming attacks and environmental hazards.

Combat encounters are designed to be dynamic, with enemy placement, mini-boss behaviors, and environmental traps requiring players to make real-time decisions while balancing offensive output and resource conservation. Strategic exploration and observation of points of interest often provide opportunities to prepare for encounters, such as finding consumables, positioning, and scouting enemy types before engagement.



## Dynamics

From the perspective of the MDA framework, the Combat System shapes the player's experience by creating meaningful choices and tension. Players feel a sense of **challenge and mastery** as they balance resource management ([arrow quantity](#), [special arrows](#)), spatial awareness (positioning and dodging), and tactical adaptation (choosing the right [path modifiers](#) and [arrow types](#)). The system encourages experimentation with different strategies, such as using [Ricochet](#) arrows to hit hidden targets, or [Wave Shot](#) to cover multiple enemies simultaneously.

The risk-reward loop is central: overextending with rare arrows may lead to running out of resources mid-expedition, forcing retreat to the hub. Conversely, careful planning and effective use of abilities and arrow types reward skillful play and provide a satisfying sense of progression and empowerment.

## Related systems

- [Health](#)

## Included mechanics

- [Shoot](#)
- [Set Arrow path modifier](#)
- [Manage Quiver](#)
- [Use Ability](#)

## Consumables

**Consumables System** – a system responsible for managing all logic related to consumable items in the game.

## Description

This system defines the types of consumables, their effects, durations, and interactions with the player and other gameplay systems. It handles:

- Buffs applied by consumables (temporary or lasting for the entire run)
- Stat modifications
- Rules for lending items to NPCs in [Missions](#)
- [Conversion](#) of unused consumables into currency
- Interaction with inventory and item usage mechanics

The system ensures consistency in how consumables behave and how their effects are applied, providing a central point for balancing and extending consumable functionality.

## Item progression

[Use Items](#) increase its effect's value as the player [explores the dungeon](#). This increased value is represented as an increase in the item's statistics and associated gold value.

This ensures that the player is rewarded if they are capable of surviving for longer. As the

enemies also increase in strength, the items will give the player stronger buffs when consumed, making them still worth the consumption even if their value is higher.

The value of items increases in three levels ( **I** , **II** and **III** ) with **I** being the lowest and **III** the highest.



## Consumable types

#TODO List of consumables

## Related Systems

- [Inventory](#)
- [Missions](#)

## Difficulty

**Difficulty System** – This system shows how the game increases in difficulty throughout the playthrough.

## Description

When the player enters an unexplored room, a number is added to a counter. The game becomes progressively more difficult as the player explores more rooms. Also, the items appearing in the rooms will become more valuable.

## Dynamics

The player must hurry before the game becomes very difficult, making it more dynamic. Players must decide whether if it is worth to explore more rooms and acquire more resources, which makes the game harder.

This gives the player an interesting risk and reward choices to make, as dying would mean loosing all items found during the run, but exploring and fighting would mean getting better items.

## Related systems

- [Dungeon Exploration](#)
- [Enemy](#)
- [Consumables](#)

## Included mechanics

- [Enter and exit the dungeon](#)

## Dungeon Exploration

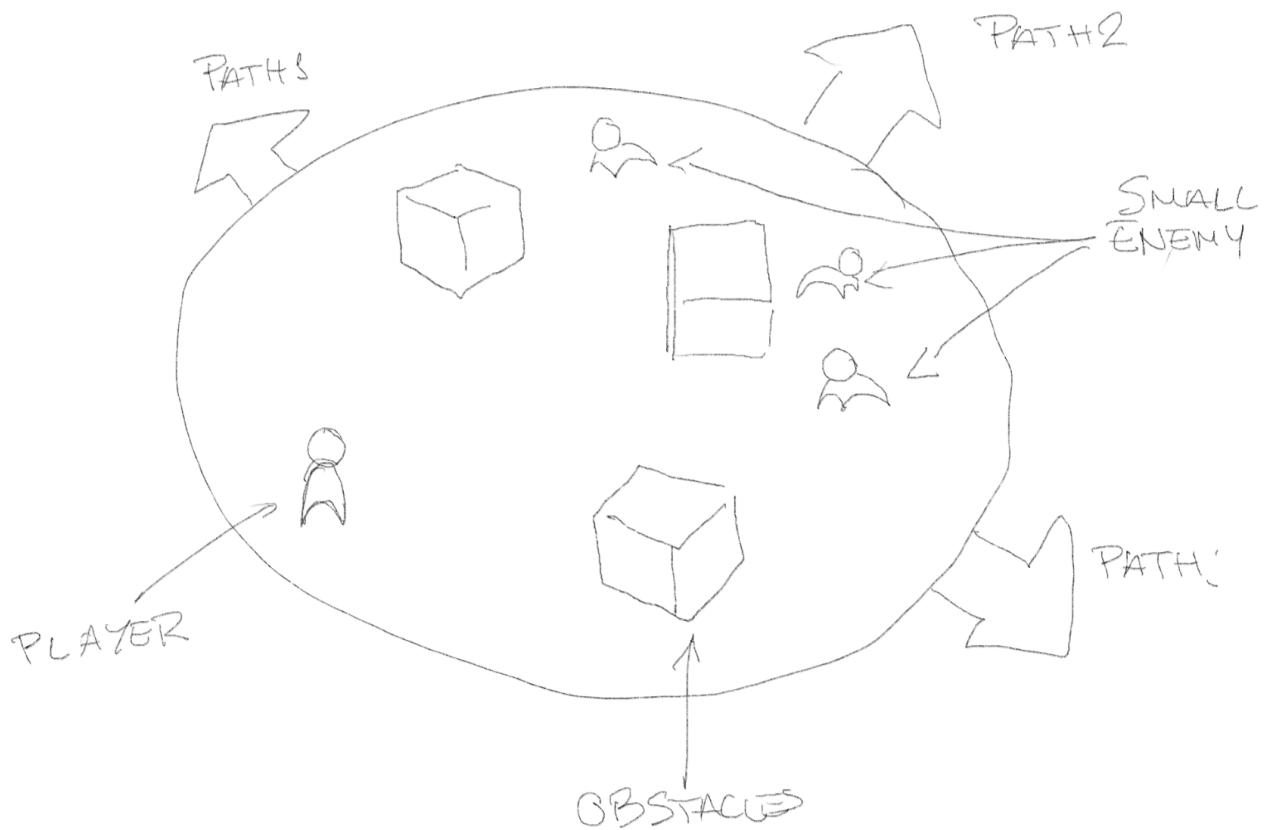
**Dungeon Exploration System** – a system that defines the layout of the dungeon and manages the player's movement between rooms and it's content.

## Description

The dungeon is composed by rooms. Each room has a number of [paths](#) that connect with other rooms (like Hades or Isaac). As the player traverses new rooms, the game becomes harder (look at [Difficulty](#)).

Rooms work like isolated scenes and are hand-crafted. Each room has [room entities](#) that the player can interact with.

Paths are predefined, although we would like to make a procedurally generated dungeon.



The dungeon could be represented as a graph, with each room being the nodes, and each path the edges:

# Random dungeon example

The diagram illustrates a random dungeon layout as a graph. The nodes are represented by colored circles: grey for 'Fight Room', orange for 'Exit', blue for 'NPC', red for 'BOSS', and green for 'Entrance'. The connections are as follows:

- The 'Entrance' (green) connects to a 'Fight Room' (grey).
- This 'Fight Room' connects to another 'Fight Room' (grey).
- This 'Fight Room' connects to an 'Exit' (orange) and an 'NPC' (blue).
- The 'Exit' (orange) connects to a 'Fight Room' (grey).
- This 'Fight Room' connects to a 'Fight Room' (grey) and an 'NPC' (blue).
- The 'NPC' (blue) connects to a 'Fight Room' (grey).
- This 'Fight Room' connects to a 'BOSS' (red) and an 'Exit' (orange).
- The 'Exit' (orange) connects to a 'Fight Room' (grey).
- This 'Fight Room' connects to an 'Exit' (orange).

```
graph TD; Entrance((Entrance)) --- F1((Fight Room)); F1 --- F2((Fight Room)); F2 --- Exit1((Exit)); F2 --- NPC1((NPC)); Exit1 --- F3((Fight Room)); F3 --- F4((Fight Room)); F3 --- NPC2((NPC)); NPC1 --- F5((Fight Room)); F5 --- BOSS((BOSS)); F5 --- Exit2((Exit)); Exit2 --- F6((Fight Room)); F6 --- Exit3((Exit));
```

## Rooms

Each room is hand-crafted; therefore, its entities are defined in the design.

Room entities are persistent. When a room is unloaded, the state of all its entities must be saved. For example, if the player kills all the enemies in a room and then exits, the enemies will still be dead when the player returns. The same applies to consumables and NPC dialogues.

¿El jugador tiene que matar a todos los enemigos para desbloquear los caminos?

#TODO

#TODO

## Dynamics

Each room will have a specialized purpose. For example, one room cannot be both a fighting room and an NPC room. This way, we can separate and encapsulate how the player interacts with each room. This allows the player to choose a path based on their desired strategy.

The player should be encouraged to clear each room before enter another one. This can be

achieved by either blocking the paths or rewarding the player with valuable items after clearing the room.

## Included mechanics

- [Change Room](#)
- [Enter and exit the dungeon](#)

## Enemy

**Enemy System** – a system responsible for managing enemy types, behaviors, and interactions with the player.

## Description

The Enemy System defines the different types of enemies that populate the game world, including small enemies, mini-bosses, and bosses. It specifies their behavior patterns, attack styles, movement logic, and interactions with the player and the environment. The system ensures that each encounter feels unique and requires players to adapt their strategy according to enemy type and behavior.

## Dynamics

Players must observe enemy patterns, react to attacks, and prioritize targets based on behavior and threat level. Variety in enemy types encourages experimentation with combat strategies, positioning, and use of abilities. Environmental interactions, such as traps or cover, further influence enemy behavior and encounter dynamics.

## Included mechanics

- [Enemy Attack](#)

## Health

**Health System** – a core combat system that manages the player's remaining vitality and determines how much damage they can take before the run ends.

## Description

Health is implemented as a bar with generally larger values (e.g. 160/250). This allows designers to choose from more damage values in enemies, therefore making health and damage easier to correct and adjust.

When enemies hit the player (through contact or projectiles), the player's health is reduced a certain amount of health points. This will be represented with the bar swiftly reducing until it's adjusted to the new value.

When the player's health reaches 0, the player dies and returns to the HUB, losing all items obtained during the run.



Health is recovered when the player returns to the HUB, by dying or [exiting the dungeon](#). Health can also be recovered by consuming items which recover the player's health.

## Dynamics

The health bar must show the player the range of error he has until his run ends because he dies. This means that when the health is full, the player must feel like he can take risks and explore. But as health decreases, the health must very clearly show the player how he can die at any moment.

Therefore, high health would make the player feel secure and adventurous, and low health bar must make the player feel scared and insecure with his possibilities.

## Related systems

- [Combat](#)

## Interaction



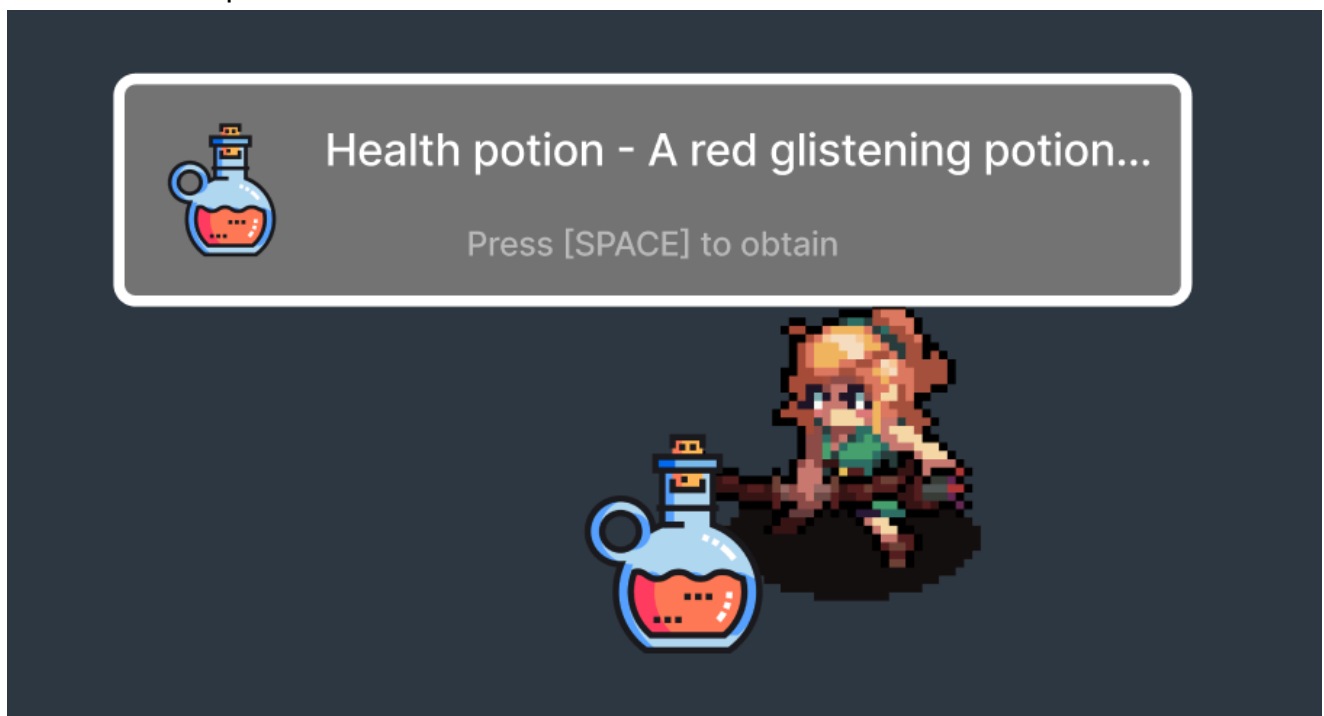
**Interaction System** – a versatile system that describes how players can interact with NPCs or objects.

## Description

Through this system, the player will be able to interact with non-player characters (NPCs) and objects.

When the player interacts with an NPC, a bubble will appear over the character. The text that the NPC is saying will be displayed in this bubble. Below the bubble, there will be one or two options for responses. Next to the bubble will be a portrait of the character.

Interacting with an object (opening a door, a crate, or entering the dungeon) works the same way, but instead of a portrait, there will be a drawing, and instead of dialogue, there will be a narrative description.



To interact with an entity the player will approach said entity and press the [interact](#) button.

## Dynamics

Having a common yet versatile system for all non-combat interactions in the game is a great way to simplify the experience for the player, reducing the amount of systems one has to learn to play the game.

The player will have a better view of the appearance of the NPC who is speaking.

The UI will be over the interactive target, making it more immersive.

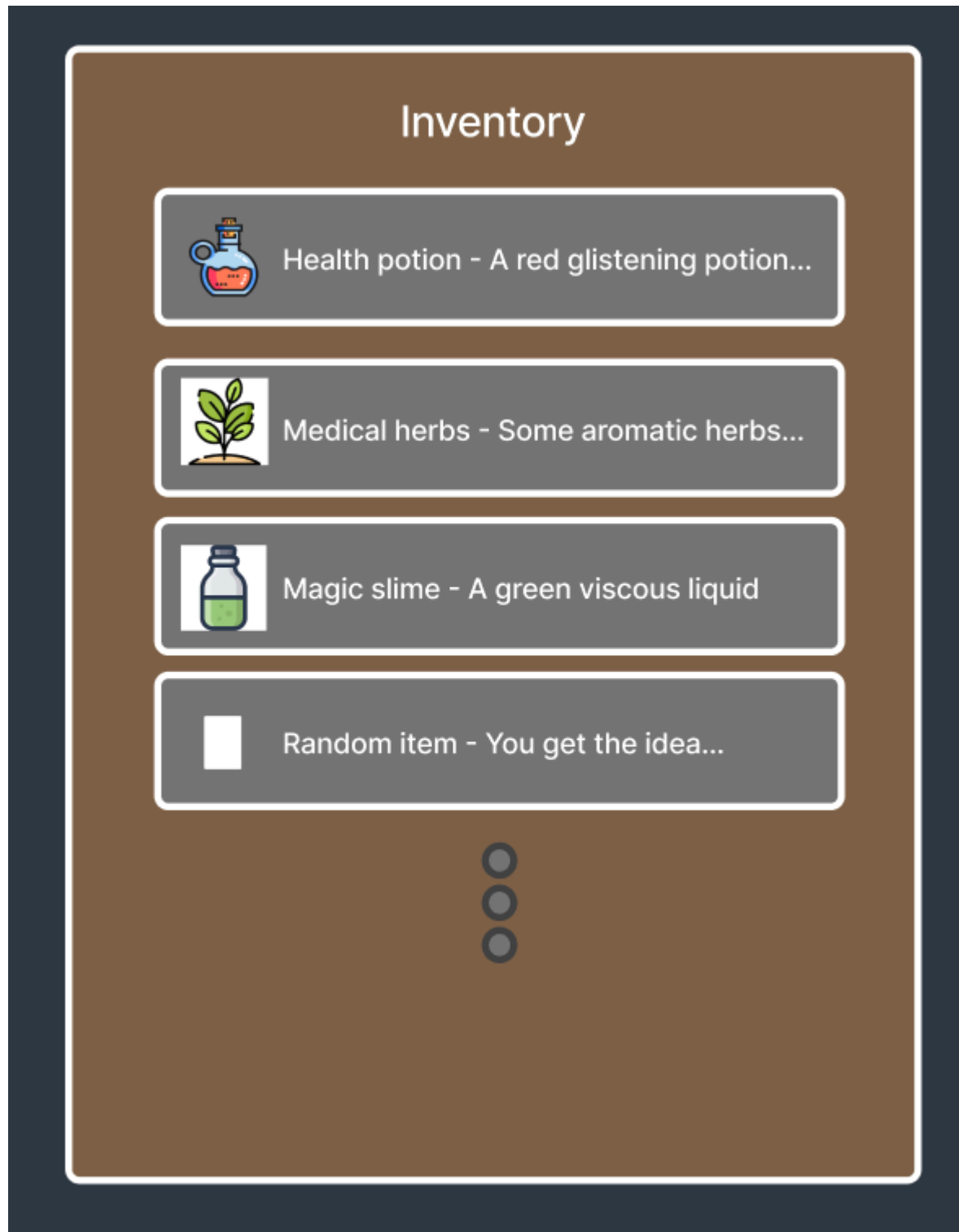
## Inventory

**Inventory System** – a system that manages the player's inventory throughout its entire lifecycle, from entering the dungeon to exiting it.

## Description

Players will be able to collect [items](#) during the run through the [interaction system](#). Items collected by the player are stored inside his inventory.

The inventory is an infinite list of items collected by the player, so the player can take as many items as they want.



All items inside the player's inventory are sold when the player [exits the dungeon](#). However, if the player dies, all items inside the inventory are lost.

## Included mechanics

- [Use Item](#)
- [Pick up items](#)
- [Sell item](#)

## Missions

**Missions System** – a system that manages mission progression and rewards players upon their completion.

## Description

Before entering the dungeon for the first time the player encounters an NPC that will give him a mission: "Enter the dungeon and come back alive".

After completing this first mission the [Shop](#) will be unlocked.

From this moment onwards the missions will vary depending on the data gathered from the runs with the objective of making the player try different types of gameplays, for example if the player is focused on recollecting items but not engaging in battle, the npc will give the player missions to subjugate monsters. The missions can go from collecting specific items from the dungeon to killing some type/quantity of enemies.

Completing a mission will grant you benefits in the shop, making upgrades and consumables cheaper, and also will unlock the different types of Arrows.

## Dynamics

El jugador asocia recompensas positivas al hacer misiones, de esa manera se fomenta hacerlas.

El jugador es animado con misiones a probar todo lo que puede ofrecer las mecánicas, para que así pueda desarrollar mejores estrategias en la dungeon.

## Examples

1. "Enter the dungeon and come back alive" - Unlocks Shop.
2. "Kill 3 enemies of type (?)" - Unlocks Type of Arrow.
3. "Collect (?) type of Item" - Unlocks Type of Consumable.
4. "Collect (?) quantity of (?) Item " - Discounts on the shop.
5. "Unlock (?) different types of areas in the same run" - Unlocks Abilities.

## Included mechanics

- [Interaction](#)

## Movement

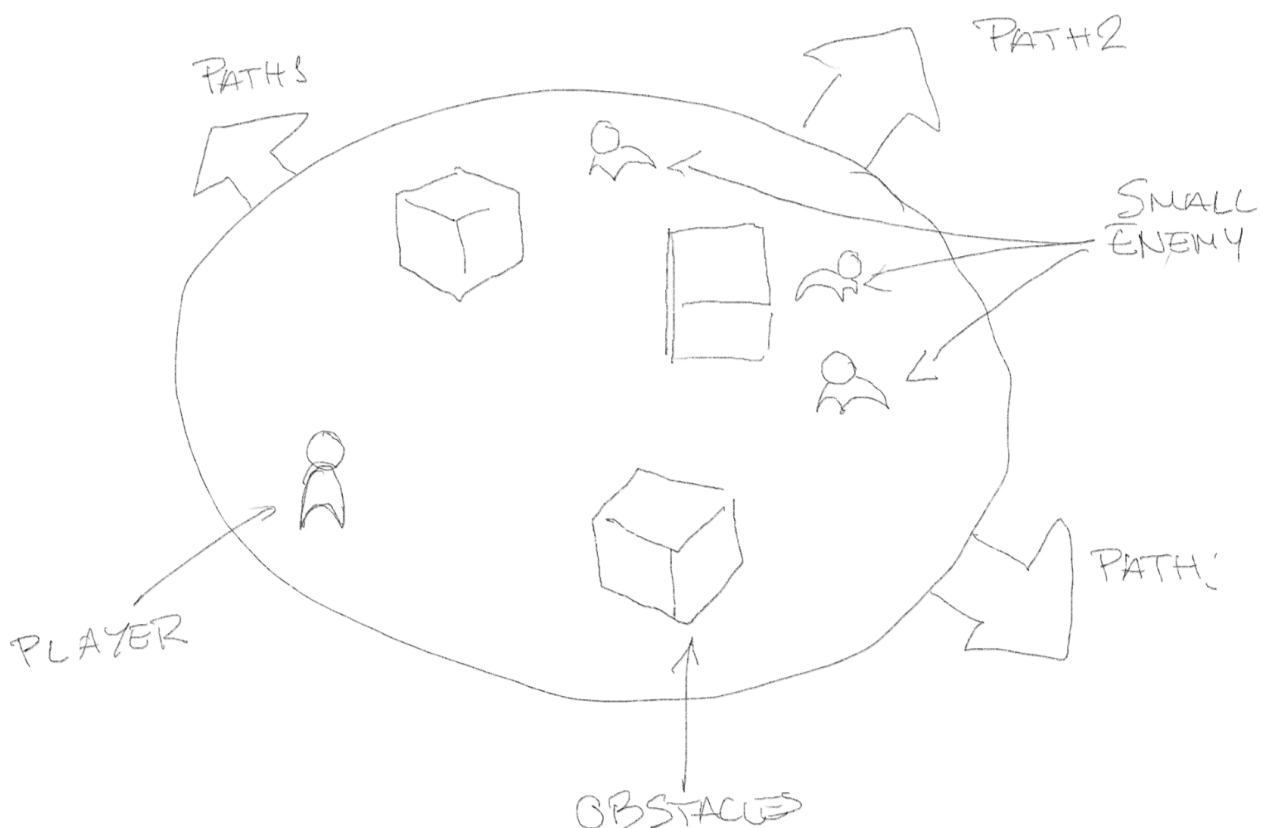
**Movement System** – a core system that governs the player's traversal through the In-game world, including running, dodging, and using abilities.

## Description

The player moves through the 2D room, in a continuous way. Rooms have a defined size and the player will not be able to move beyond the room's edge.

When the player traverses one of the room's paths, they will go to another room.

The player will collide with static objects in the room (trees, structures, stones, ...), but will be able to move through enemies and other dynamic entities such as objects or NPCs.



Player movement is not accelerated, it is a constant velocity. Due to the keyboard limitations (the most commonly used input in web), the direction in which the player moves is discrete, and only within the values of 8 directions.

Player movement may be altered through the different player abilities.

## Dynamics

The player must feel like his character is completely under his control, with precision and accuracy. Movement must feel responsive and fast, but especially satisfying, as it will be what the player does most often.

## Included mechanics

#Todo Use Ability

## Shop

**Shop System** – a system that manages the game's economy and how the player can earn and spend coins.

## Description

Before a run, the player can buy the equipment he wishes to use with coins. The stronger the equipment, the more coins needed to buy it.

The player can buy only 2 types of items for the next run:

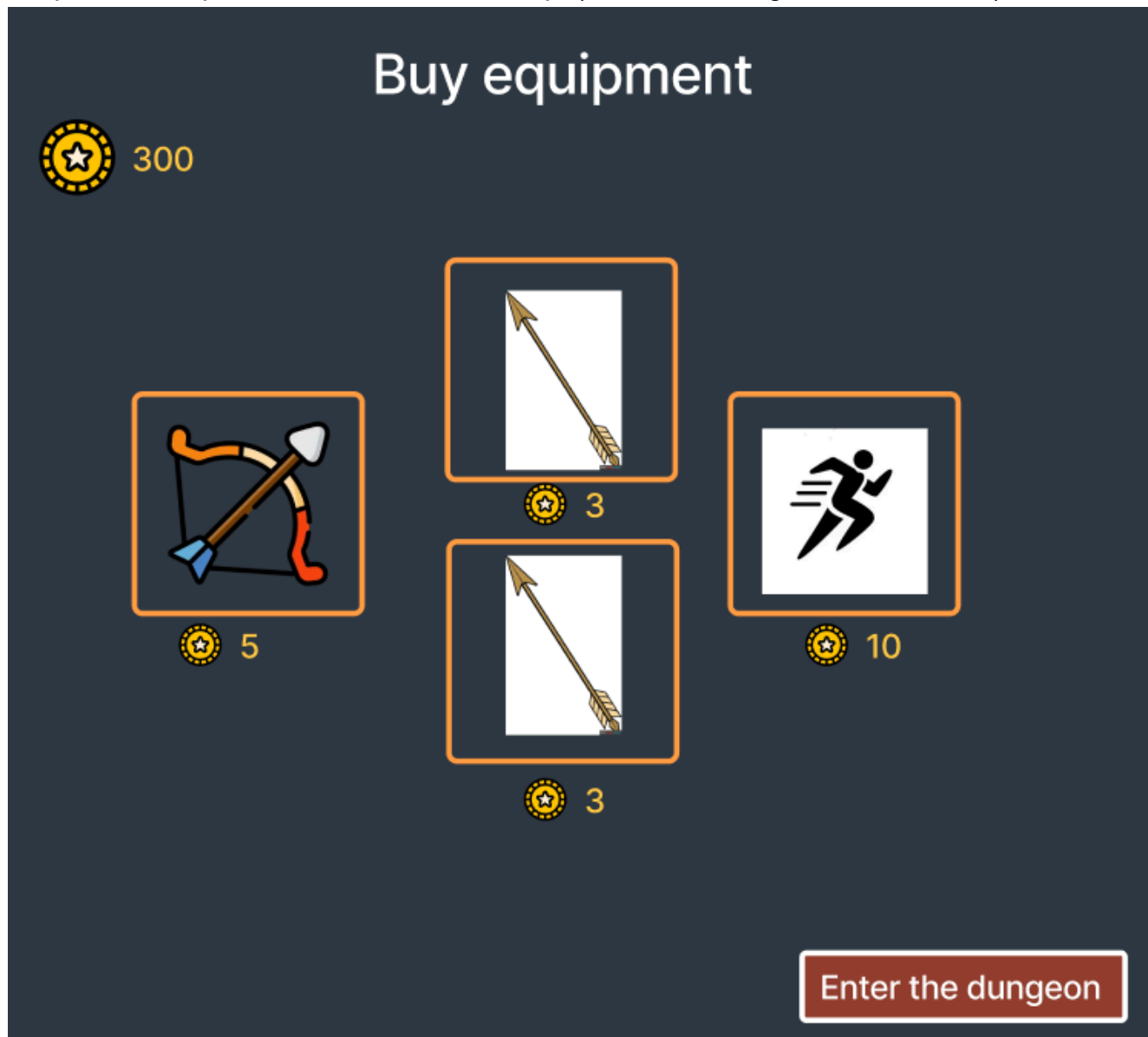
- Bows ([Arrows trajectories](#))
- Arrows (Two types of [arrows with special effect](#))

## Coins

Coins are persistent between runs, meaning that they are not lost when the player dies or begins a new run. Coins are obtained by selling [items](#) when [exiting the dungeon](#).

## Prototype

Graphical example to understand the concept (Not actual design, not even close):



## Dynamics

As the player obtains items and improves, he will also gain more gold, allowing him to [obtain better items](#).

If the player runs out of gold, he will always have a basic loadout for free with the most basic items. This ensures that the player can always recover himself from big losses and represents the base point of the equipment progression.

The player is confronted with whether it is worth spending more for better or more custom items, or save gold for another future run.

When the player enters the dungeon, he risks losing all of his equipment if he dies.

Therefore, the player must be careful to buy items which allow him to survive more.

This also enhances the tension throughout the run, as dying would mean losing the items bought at the beginning.

## Included mechanics

- [Sell item](#)

## Mechanics

Below, all game mechanics are described

## Combat

---

### Manage Quiver

**Manage Quiver** – a mechanic that allows the player to manage and switch between different types of projectiles.

### Description

The player can equip two different types of arrows in the quiver, each with unique properties that affect combat strategy. Arrows are consumable items: when the player shoots, the corresponding arrow is removed from the inventory.

The player can quickly change arrows in the middle of combat to use both arrows properties using [change arrow key](#).

Arrows are a limited resource as the player can only have a certain number of arrows in their inventory. Arrows must be picked up after being used, making shooting arrows more than just shooting everywhere.

Stronger arrows are more limited in quantity, requiring the player to carefully manage resources and choose the right type of projectile for each encounter. This mechanic encourages strategic thinking and careful planning during exploration and combat.



## Arrow Effects

### Fire Arrow

The *fire arrow* deals `param` every `param` seconds during `param` time when impacting an enemy. When combined with other arrow effects, the *fire effect* will, in some way, enhance the damage dealt to the enemy.

The fire arrow is has a `param` seconds of cool down per shot.

### Slime Arrow

The *slime arrow* explodes in an area of slime (with `param` diameter) when impacting an enemy , which slows down enemies `param` % for `param` seconds. When combined with



other effects, the *slime effect* will add to the duration time of other effects.  
The slime arrow is has a `param` seconds of cool down per shot.

## Gas Arrow

The *gas arrow* explodes into a cloud of a poison when impacting an enemy. All enemies inside the cloud receive `param` damage every `param` seconds, the longer the enemies are under the effect of the poison, the more damage it deals.  
The gas arrow has a `param` seconds of cool down per shot.

## Grass Arrow

The *grass arrow* immobilizes the enemy hit by the arrow (grass grows and traps the enemy). The enemies will not be able to move for `param` seconds.  
The grass arrow is has a `param` seconds of cool down per shot.

## Combined effects

Effect combination can happen in any order, meaning that effects do not need to be applied in an specific order for the special effect to happen.

### Fire + Slime

When combining *fire* with *slime*, the fire effect is prolonged for `param` seconds and damage is increased by `param` %.

### Fire + Gas

When combining *fire* with *gas*, an explosion occurs, dealing `param` damage to all enemies in the area and setting them on fire.

### Fire + Grass

When combining *fire* with *grass*, the fire expands through the grass, setting all enemies on fire.

### Slime + Grass

When combining *slime* with *grass*, the grass effect duration is enhanced by `param` % and the grass effect will affect all enemies in the area of effect of the slime.

### Slime + Gas

When combining *slime* with *gas*, the gas effect will stick to the enemy even if it is outside the cloud for `param` seconds.

## Gas + Grass

Does nothing special, their basic effects combined is already powerful enough.

## Related Systems

- [Combat](#)

## Set Arrow path modifier

**Set Arrow Path Modifier** – a mechanic that allows the player to alter the trajectory and behavior of projectiles after they are fired by selecting special type of bow.

## Description

The player [can equip](#) one active arrow path modifier (bow type) before entering the magical realm.

This modifier changes the flight pattern of all fired projectiles, overriding the default arched trajectory and adding unique behavior.

Modifiers can alter the number of projectiles, their spread, bounce behavior, and travel distance. It is used to adjust the area of effect, control coverage, or introduce tactical variety in ranged combat.

Only one path modifier can be active before [enter the dungeon](#), encouraging players to adapt their loadout to different enemy types and environments.

## Available Modifiers

- [Hunter's Bow – Triple Shot](#)
- [Falcon Bow – Long Shot](#)
- [Rebound Bow – Ricochet](#)
- [Piercer Bow – Piercing Shot](#)
- [Seeker Bow – Homing Shot](#)
- [Graviton Bow – Magnet Shot](#)
- [Specter Bow – Phantom Shot](#)

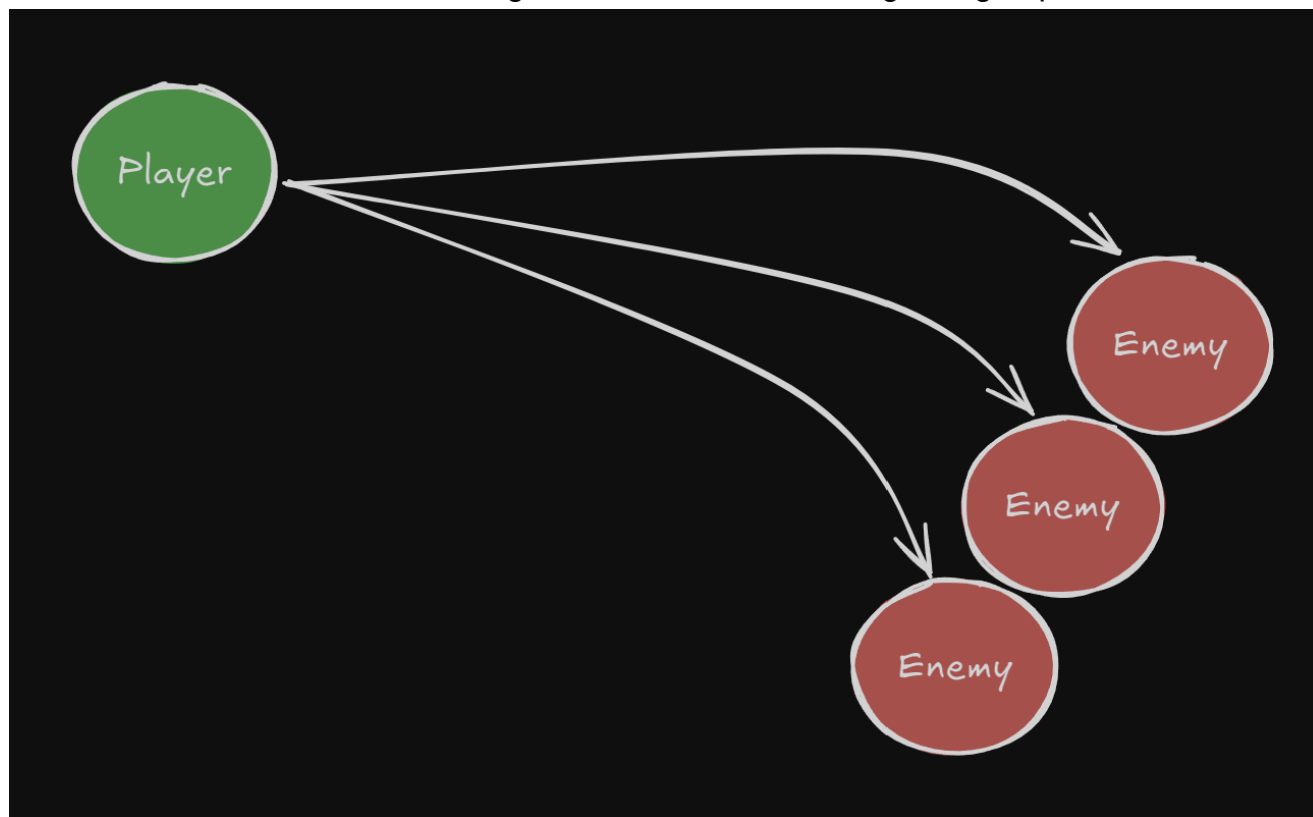
## Hunter's Bow – Triple Shot

Fires **three arrows** simultaneously in a **spread formation** (angled slightly outward).

Each individual arrow deals **reduced damage**, but the total output can be high if all arrows

hit the target.

This bow excels in close to mid-range combat, ideal for dealing with groups of enemies.



## Falcon Bow – Long Shot

**Increases the initial velocity** of the arrow and reduces its trajectory curvature, allowing it to travel **much farther** in a **straighter line**.

This modifier is best for **long-distance precision**, letting skilled players snipe enemies before they can close in.



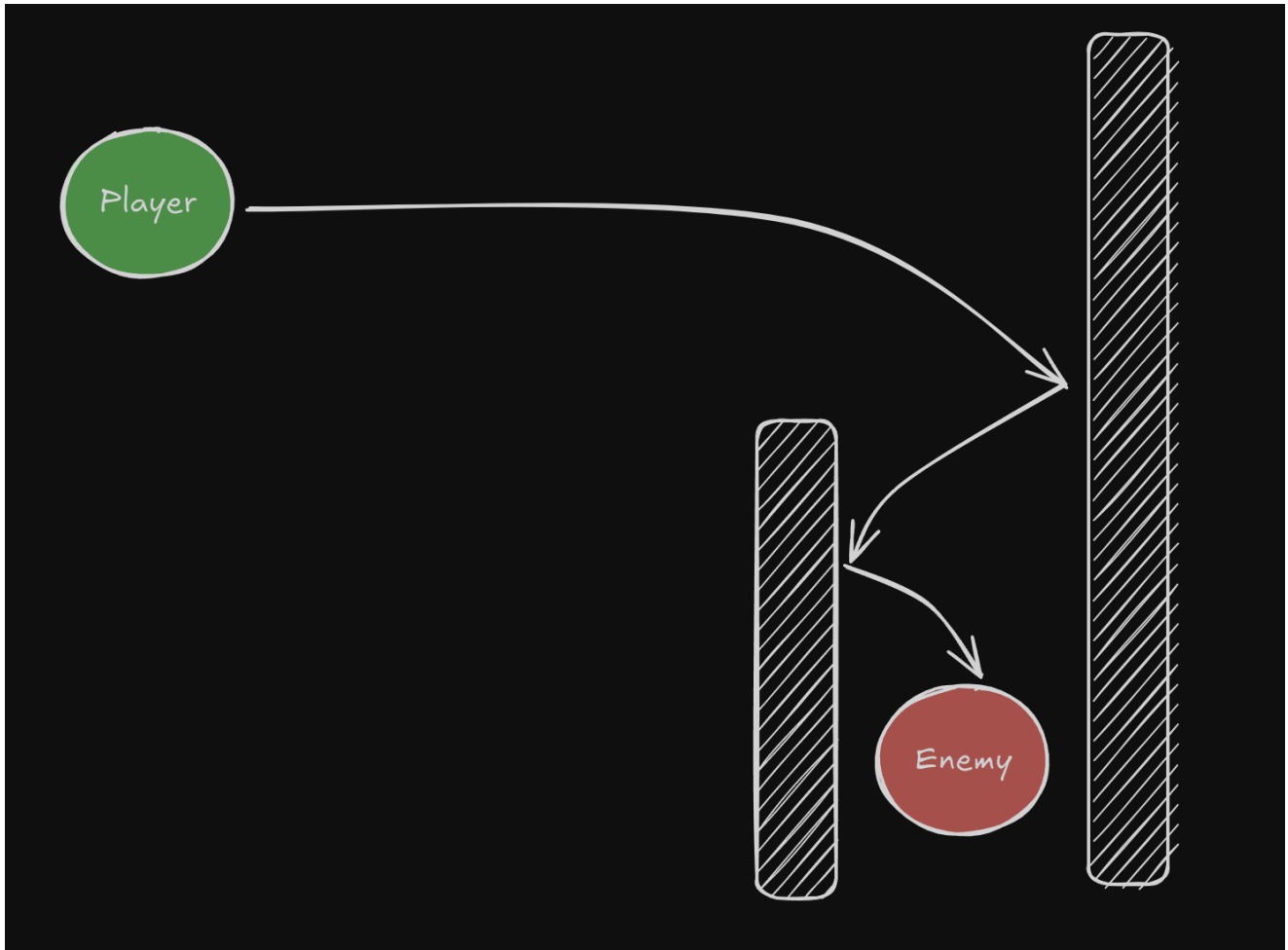
## Rebound Bow – Ricochet

Arrows gain the ability to **bounce** off surfaces (walls, objects, possibly enemies) **multiple times** before losing momentum.

Each bounce slightly reduces velocity and damage.

Can be used to hit enemies hiding behind cover or around corners, rewarding players who

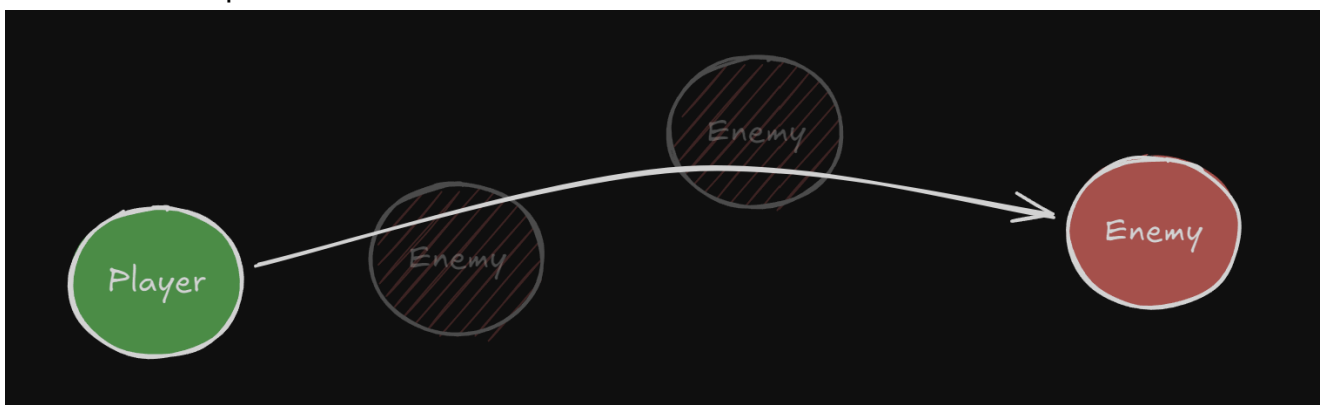
master geometry and angles.



## Piercer Bow – Piercing Shot

Arrows **do not stop upon impact** and can pass through multiple enemies in a line. This makes it especially effective against **dense enemy formations** or narrow corridors. Each consecutive hit deals **reduced damage**.

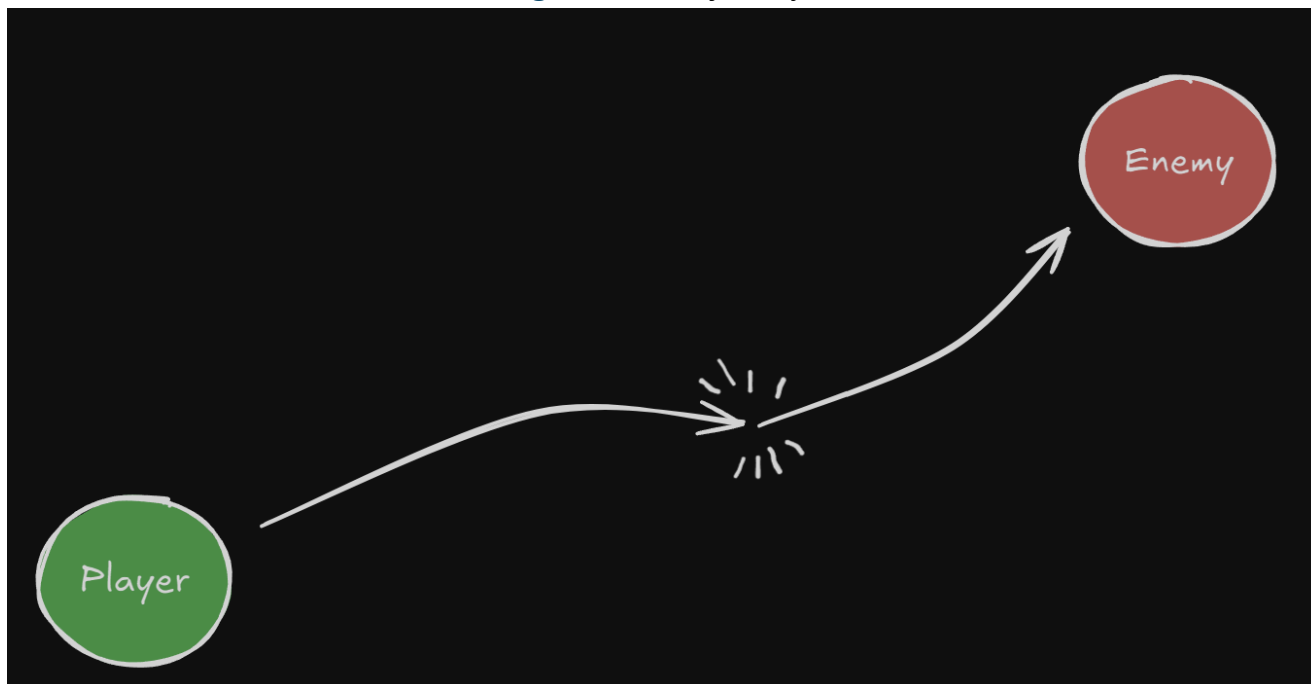
Implementation-wise, this can be achieved by temporarily disabling collision for a few frames after the first impact.



## Seeker Bow – Homing Shot

After an arrow **loses altitude** and would normally **hit the ground**, it **locks onto the nearest target's last known position** and performs an additional **dash** toward it before stopping.

The trade-off is **reduced base damage** due to trajectory correction.

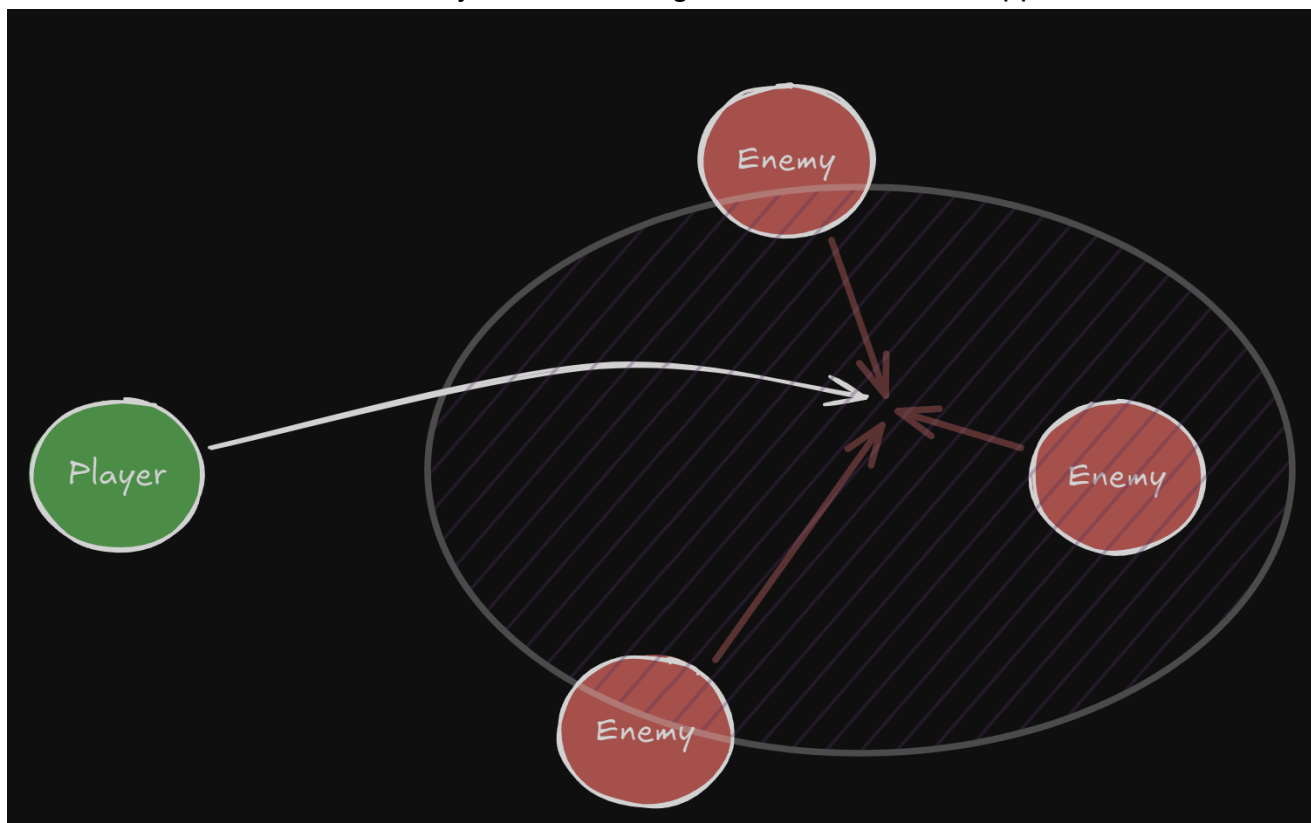


## Graviton Bow – Magnet Shot

After landing, the arrow **generates a magnetic field**, slowly pulling in nearby **enemies, consumables, or dropped arrows**.

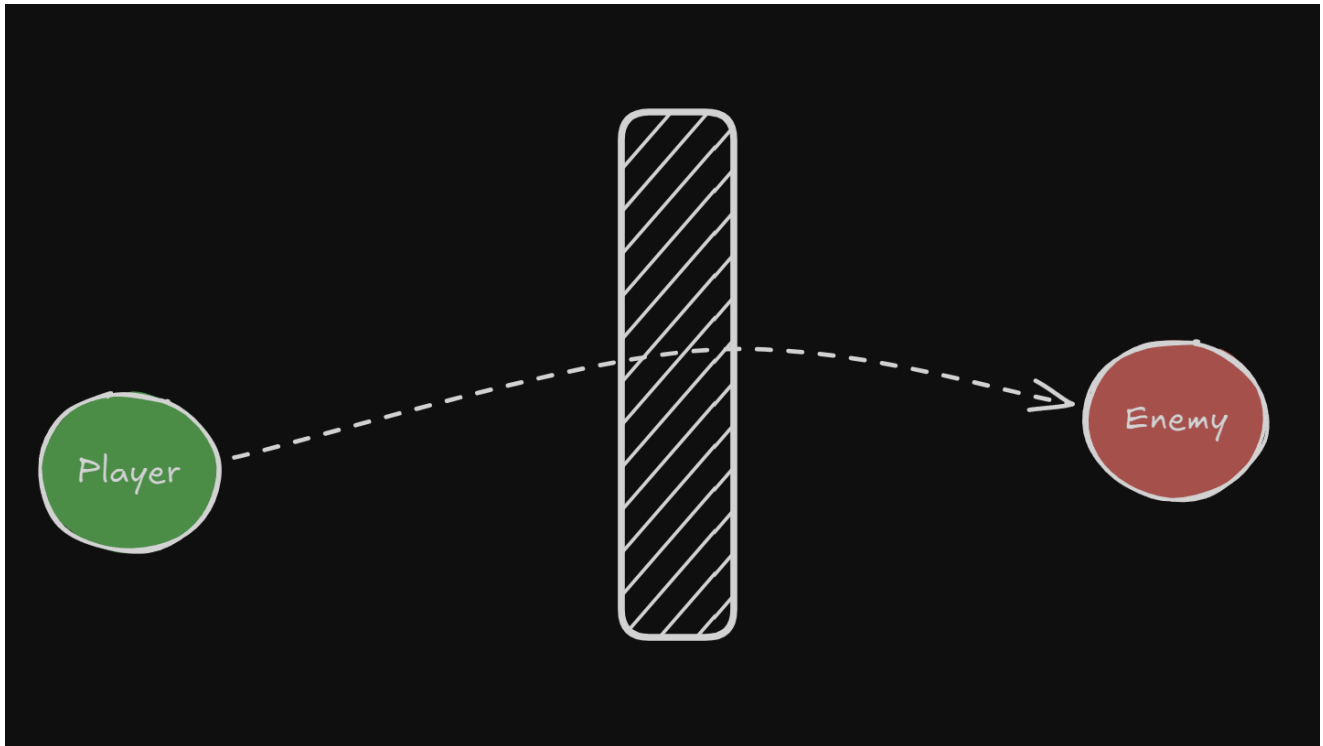
This allows the player to **control enemy positioning** and simplify item retrieval.

Perfect for crowd control or utility builds, offering both offensive and supportive benefits.



## Specter Bow – Phantom Shot

Arrows **phase through obstacles**, passing through **one wall or shield** before dissipating. This enables the player to **hit enemies hiding behind cover**, introducing a tactical advantage in complex environments. However, the **ammo count is reduced**, so player can't just spam all his arrows through the walls.



## Related Systems

- [Combat](#)

## Shoot

**Shoot** – a mechanic that allows the player to perform ranged attacks using a bow. It is used for engaging enemies from a distance, applying status effects, and interacting with certain elements of the environment.

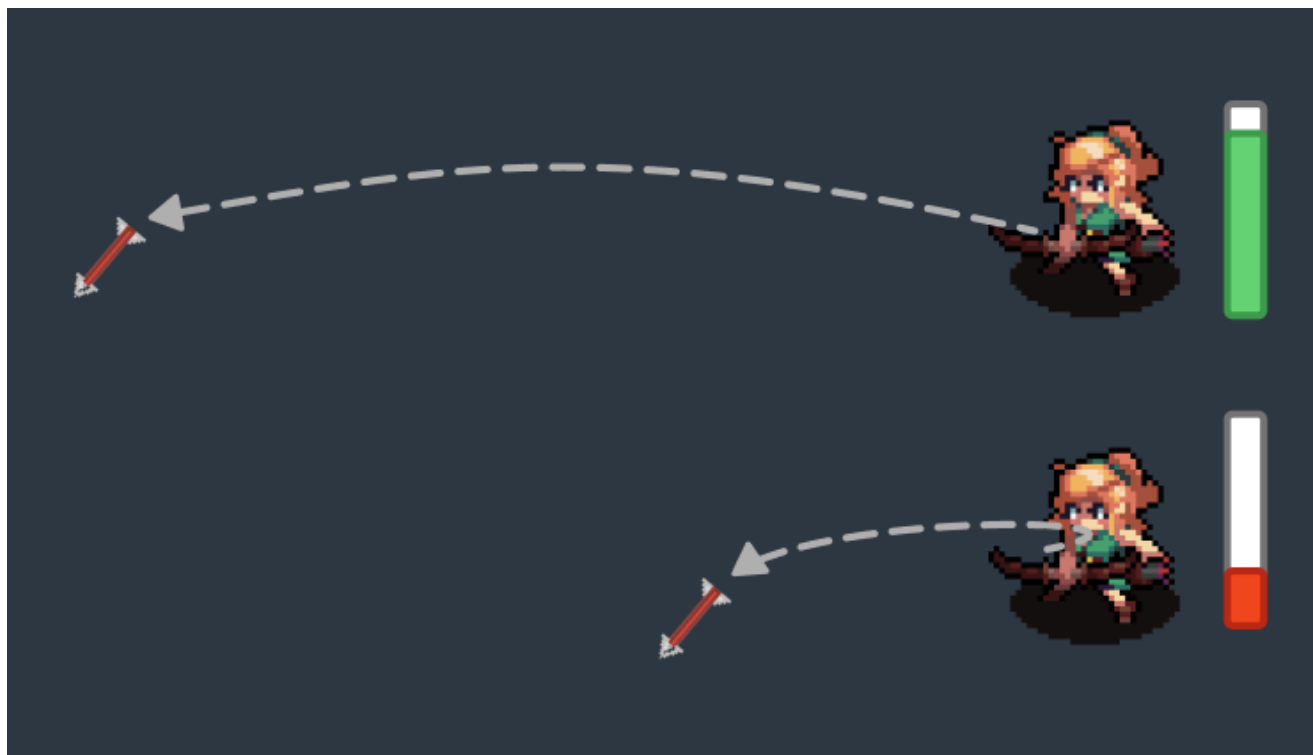
## Description

The player aims [using his input device](#) and presses the [shoot button](#) to release the [equipped projectile](#) in the [chosen direction](#).

To control the projectile's force, the player must **hold** the shoot button for a certain duration. The longer the button is held, the stronger the shot will be.

Projectiles (arrows) follow an arched trajectory. The curvature of the arch depends on the force applied by the player. If the player barely charges the shot, the arrow travels in a steep

arc with limited distance. If the player charges the shot to maximum, the arrow flies in a near-straight line\*, covering more distance and having a flatter trajectory.



When an arrow is shot, it is removed from the player inventory. To recover it, the player must go to where the arrow has landed to pick it up.

## Related Systems

- [Combat](#)

## Use Ability

**Use Ability** – a mechanic that allows the player to use a special ability that affects movement. It is used to enhance exploration and mobility during fights.

## Description

Before each expedition, the player can choose one active ability from a set of available abilities. Each ability is tied to a specific item that can be found in the magical realm, making exploration a key part of unlocking new movement options.

Abilities modify the player's movement in a unique way, for example by granting temporary invulnerability frames, or boosting speed. During the expedition, the selected ability can be activated by [pressing the designated ability](#) button.

This mechanic encourages strategic planning before entering the magical realm and adds depth to traversal during exploration. Abilities are limited to one active choice per expedition, making the selection meaningful and impactful.

Every ability has a cooldown and a duration of its effect, meaning that the player cannot spam the use of abilities. Also, abilities might have a different duration until their effect disappears, for some abilities this might be instant, but for others it could mean that the effect last a couple of seconds.

## Available abilities

#TODO List of abilities

## Related Systems

- [Movement](#)

## Enemies

### Enemy Attack

**Enemy Attack** – a mechanic that allows enemies to perform offensive actions against the player. It is used to challenge the player, creating a direct threat and the risk of losing resources, such as health, forcing the player to react, adapt, and manage their survival during encounters.

### Description

Each enemy executes attacks according to its **behavior pattern** and attack logic. Attacks can be only ranged projectiles, or area-of-effect abilities, depending on the enemy type.

The timing, range, and effect of each attack are defined by the enemy's AI and may vary based on current [difficulty level](#) or environmental conditions. Attacks require the player to dodge to avoid taking damage.

## Related Systems

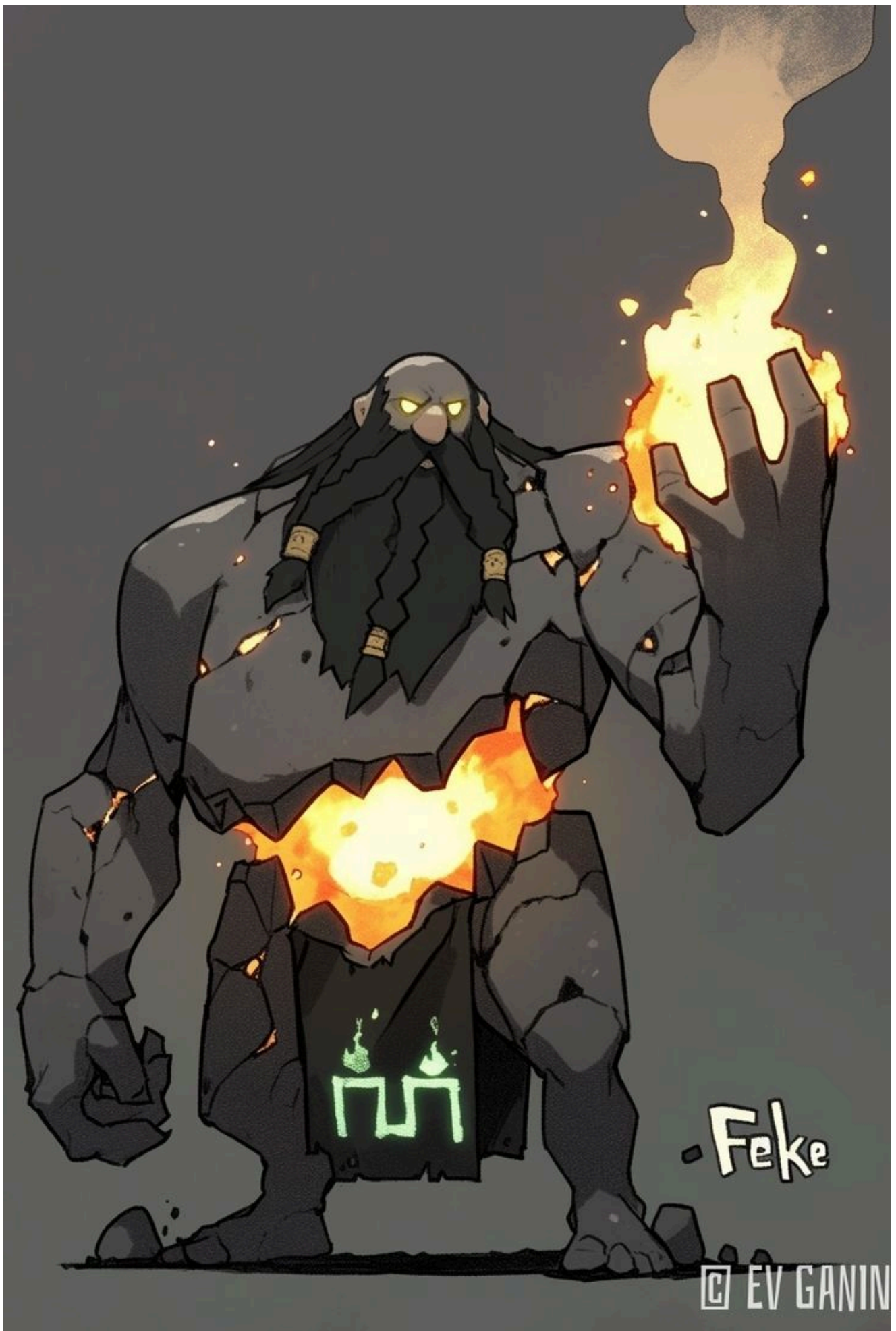
- [Combat](#)
- [Enemy](#)
- [Boss Fight](#)



## Enemy Types

### GOLEM





- fire magic attacks

- low attack range
- mid attack speed

SLIME



- slime type attacks
- high attack range
- low attack speed

DRIADE





V. LAWEYO



gas and plant attacks  
medium attack range  
mid attack speed

## Exploration

---

### Change Room

**Change Room** – a mechanic that allows the player to move between rooms.

### Description

Each room is hand-crafted and contains one or more doors or paths. When the player [interacts with a door](#), they are transported to the connected room. The player spawns at the corresponding entry point in the new room, typically the path that leads back to the previous room.

## Related Systems

- [Dungeon Exploration](#)

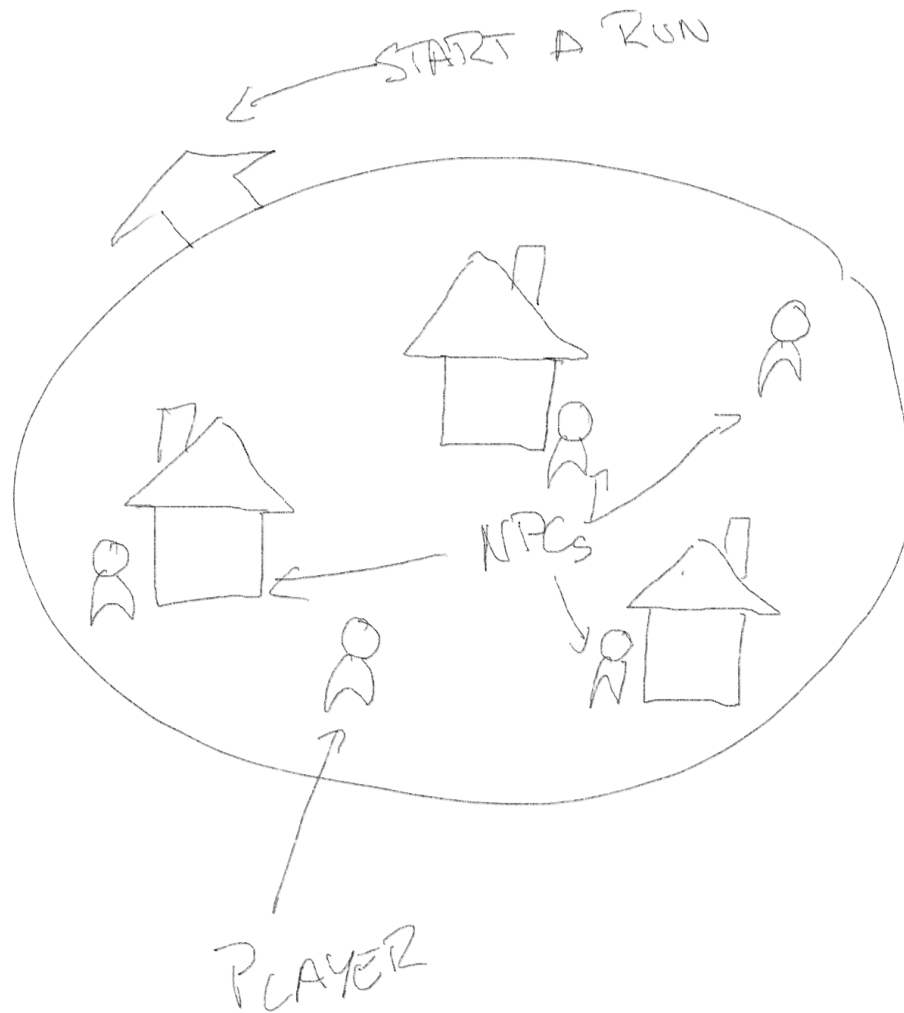
## Enter and exit the dungeon

**Enter and Exit the Dungeon** – a mechanic that allows the player to enter and leave the dungeon freely, avoiding any soft-blocks.

## Description

Players will enter and exit the dungeon through specific entrance and exit rooms. These rooms will only be used for entering and exiting the dungeon and will not contain any [room entities](#)

The player can only enter the dungeon from the HUB, and must always have bought a loadout in the [equipment shop](#) before entering. Once inside, the player will appear in one of the many entrances to the dungeon.



When the player exits the dungeon through the exit room, he will return to the HUB and sell all objects in his inventory [Sell item](#).

There will be x number of entrances and exits. One of the entrances is chosen at random when the player enters the dungeon. The player can exit the dungeon through any exit.

## Related Systems

- [Dungeon Exploration](#)

## Items

---

### Item List

- - Attack -

- - Health -
- - Attack Speed -
- - Movement Speed -
- - Critical Chance -
- - Critical Damage -
- Dungeon Escape -  
(Damage based obtained through battle? and life and movement through recolection?)

Item type	Lvl1	Lvl2	Lvl3	Rare
Attack	+50% atk dmg 15 sec	+100% atk dmg 20 sec	+200% atk dmg 35 sec	+ 300% atk dmg 60 sec
Health	+ 5 hp	+15 hp	+ 35 hp	+ 75 hp
Atk Speed	x1,25 atk speed	x1,5 atk speed	x2 atk speed	x3 atk speed
Movement speed	x1,10	x1,30	x1,60	x2,5
Critical Chance	+ 5%	+15%	+30%	+60%
Critical Damage	+30%	+70%	+150%	+200%

Dungeon Escape its a really rare item that allows the player to the outside of the dungeon without having to find an exit area.

## Pick up items

**Pick Up Items** – a mechanic that allows the player to collect items found within the dungeon.

## Description

When the player approaches an [Use Item](#) and presses the [interaction button](#), the [Interaction](#) will pop up, showing basic information of the item and asking the player if they want to pick it up or not.

If the player decides to pick it up, the item will get added to the item list inside the player's inventory.





Health potion - A red glistening potion...

Press [SPACE] to obtain



## Related Systems

- [Inventory](#)

## Sell item

**Sell item** – This mechanic describes how the player earns money by selling items.

## Description

When the player exits the dungeon, all of their [items](#) will be sold and converted into coins. Coins are the only persistent item throughout runs, and can be used in before the run to buy equipment (Look at [Shop](#)).

If the player dies, all items in his inventory are lost.

## Related Systems

- [Inventory](#)
- [Enter and exit the dungeon](#)

## Use Item

**Use Item** – a mechanic that allows the player to use a consumable from their inventory.

## Description

The player selects a consumable from the [Inventory](#) and activates it. Using the item applies its effect, which can be a temporary buff, a stat increase, or a special action depending on the item type.

## Related Systems

- [Inventory](#)
- [Consumables](#) (for item logic and effects)