

Mythgrove: Festival of Yew - Game Design Document



Mythgrove: Festival of Yew

By: Jason Skillman, Joel Hanson, Kyle Gray, Amanda Coler, Bella Weikman,
Juan Santos and Christian Martin

Table of Contents

Game Description.....	5
High Concept Statement.....	5
Genre	5
Target Audience	5
Target Platform	5
Development Platform	5
Target Rating.....	5
Length of Play.....	5
Core Game Mechanics	6
Player Archetypes	6
Reward Schedule	7
The Process.....	8
Confluence	8
Jira	9
Trello	10
GitKraken	10
Flow Charts & Diagrams.....	11
Concept Art.....	14
Early & Alpha Concept Art	14
The Hub.....	16
Player	18
NPS's	19
Monsters.....	20
Props	21
Color Palette	22
Opening Cinematic.....	23
Prototype	23
Final.....	27
Game Stages.....	31
Prototype	31
Hub.....	31
Dungeon.....	33

UI	35
Alpha	41
Hub.....	41
Shops.....	44
Dungeon.....	46
Combat.....	56
Networking.....	58
UI	59
Beta	60
Hub.....	60
Dungeon.....	65
UI	70
Release	73
Hub.....	73
Dungeon.....	79
Mechanics.....	82
Dungeon Generator (Room Builder).....	82
Rooms	86
Monsters.....	87
Weapon Generation	90
Loot Generation	92
UI.....	95
Wireframes	95
Main Menu.....	100
Credits	102
Settings.....	103
Player Inventory & Equipment.....	104
Interactions	105
Dialogue	107
Store UI	110
Sprites & Assets.....	112
Assets	116
Font	116

Sound & Effects.....	117
Particle Effects	118
Environmental Art.....	119
Swords	123
Character Models.....	125
Player	125
NPC Models.....	127
Mamo.....	127
Seawee.....	129
PikPok.....	130
Randy	131
Monsters.....	132
Vineman.....	132
Bugman	134
Boss	135
Animations & Rigging.....	136
Narrative Design.....	139
Backstory.....	139
Dialogue	139
Conclusion.....	142
Player Feedback	142
Post-Mortem.....	145
Amana Color.....	145
Jason Skillman.....	147
Joel Hanson	148
Isabella Weikman.....	149
Juan Santos	150
Kyle Gray	151
Christian Martin	153
Technical Limitations	154
Credits	154
Link.....	156

Game Description

High Concept Statement

Mythgrove: Festival of Yew is a rogue-like dungeon crawler game where the player discovers a secret forest with a dungeon entrance. The player soon realizes that the legends of the yearly dungeon festival is real and enters to meet different spirits and adventures. The player is the only human in this festival and must attend the sport of dungeoneering to enhance their fame and experience. Through a mossy dungeon terrain, the player will have to rely on their skill in battle to enhance their weapons and kill the boss to prove their worth in the festival.

Genre

Rogue-like Dungeon Crawler

Target Audience

12-17+ All Genders

Target Platform

PC

Development Platform

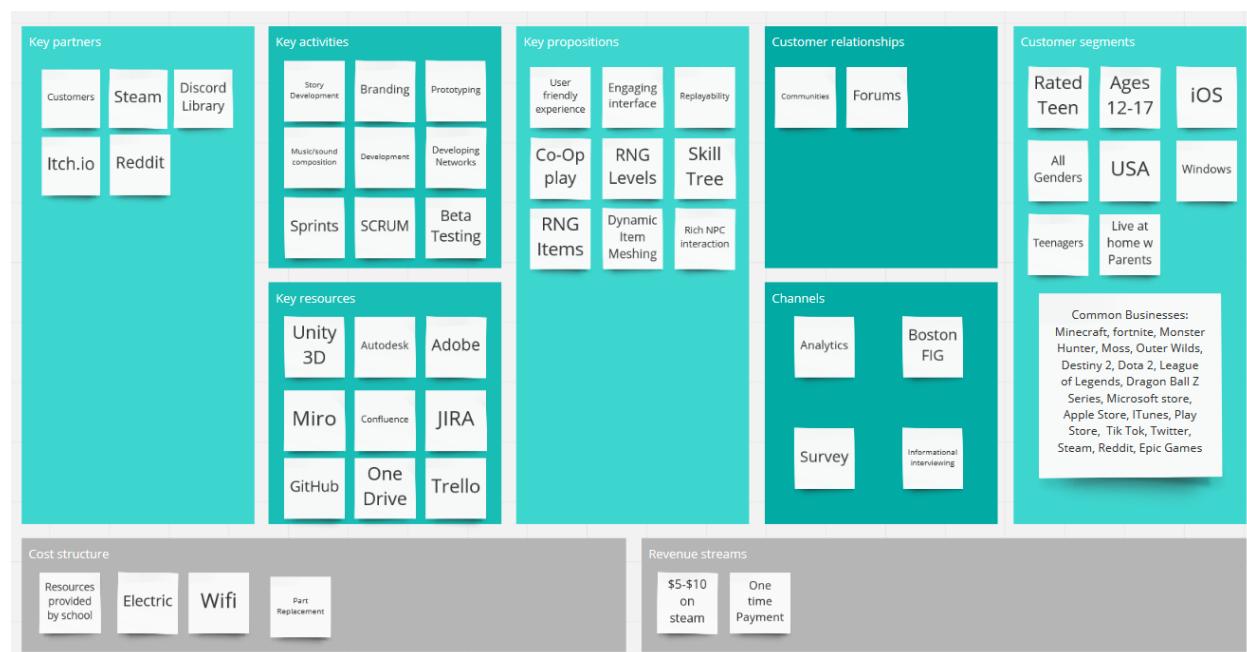
Unity

Target Rating

T for Teen

Length of Play

3 hours



Core Game Mechanics

These are the three unique ideas that make our game stand out.

1. Weapon Modifiers

- a. Based on the weapon's rank, each weapon will receive a certain number of modifiers that change the way the weapon works. This can be done by increasing the damage, having it shoot projectiles, increase player movement speed, etc.

2. Weapon Generation

- a. Each weapon is generated with random stats and meshes based on the weapon parts that were randomly chosen by the loot generator system.

3. Room Builder

- a. Each time a player enters the dungeon, the dungeon is randomly built based on room pieces. Each room that shares a matching pivot point attaches and builds off those points, creating a dungeon levels, based on a range of desired rooms.

Player Archetypes

(How different player types can affect the gameplay)

1. The Achiever

- a. They wish to receive all the achievements in the game, gain rewards, increase score
- b. The coin currency acts to gain access to more rewards & content in the game. The dungeon system is a way for players to explore, collect and progress to the boss room.

2. The Explorer

- a. They wish to explore the whole game
- b. The dungeon is randomly built each time. The player will always have to fight a series of monsters while trying to search for the boss room.

3. The Socializer

- a. Players who wish to socialize with other players
- b. Players can add friends and invite friends to the dungeon and play.

4. The Killer

- a. Players who enjoy killing things
- b. Players must kill the boss in order to exit the dungeon. There are many other monsters in the dungeon that have the chance of dropping more items that could enhance the player's ability to defeat the boss.

Reward Schedule

Variable-Ratio Schedule <ul style="list-style-type: none">• Response is reinforced after an unpredictable number of responses<ul style="list-style-type: none">◦ Examples: Gambling, lottery Games◦ RNG Item Drops◦ RNG Levels◦ Lucky Wheel◦ Pets (low)	Variable-Interval Schedule <ul style="list-style-type: none">• A response is rewarded after an unpredictable amount of time has passed<ul style="list-style-type: none">◦ Examples: Giving a pellet to a rat follow a one-minute interval; a second pellet for 5 minute interval; and a third pellet after a 3 minute interval with this pattern as a continuation.◦ RNG Item Drops◦ RNG Levels
Fixed-Ratio Schedule <ul style="list-style-type: none">• A response is reinforced only after a specified number of responses<ul style="list-style-type: none">◦ Examples: Giving a pellet to a rat after it presses the bar 5 times◦ Rare drop from boss◦ Bosses drop skill points?◦ Mobs drop some form of item	Fixed-Interval Schedule <ul style="list-style-type: none">• Responses only rewarded after a specified amount of time has elapsed<ul style="list-style-type: none">◦ Examples: Reinforcing a rat with a pellet after the first bar press after a 30 second interval has elapsed.◦ Scroll drops

The Process

Confluence

Game Design Document

Created by Amanda Coler Last updated May 06, 2020 • 3 min read

Game Engine: Unity 2019.3.1f1

Version Control: GitHub

Process Tools: JIRA, Confluence, Trello, Miro, Discord

Target/Demographic:

- **Gender:** All
- **Ages:** 12-17+
- **Rated:** T for teen
- **Location:** USA
- **Language:** English (Spanish, French (common))
- **Price:** \$5-\$10
- **Customer Interest:** Rouge-like, Dungeon Crawlers, Team-building games, RNG generation, Crafting, Unique Art,
- **Customer Systems:** Windows & Mac (mobile & Linux?)
- **Customer Business Interactions:** Minecraft, Fortnite, Monster Hunter, Moss, Outer Wilds, Destiny 2, Dota 2, League of Legends, Dragon Ball Z Series, Microsoft store, Apple Store, iTunes, Play Store, Tik Tok, Twitter, Steam, Reddit, Epic Games
- **Stage of Life:** Kids, Teenagers, Living at home with Parents, Middle School/High School

Story Notes/Dialogue Options

Created by Christian Martin
Last updated Aug 18, 2020 • 3 min read

- "Browse until your hearts content!"
- "My wares are yours for the taking if you have the coin"
- "What wonders can I bring you today?"
- "Have you seen my shop lately? Just restocked!"
- "If you don't have the cash for it, leave, and come back when you do."
- "No coin? Come back when you do, my items will be available!"
- "Just what you were lookin for, Huh?"
- "Ohhh you're gonna enjoy that one!"
- "Stocked and ready to rock your world!"
- "Step right up, Step right up! The wheel doesn't hold a grudge!"
- "Spin! Spin! Spin! Where's she gonna go? Only luck will know!"
- "My, My! What a lucky person I've happened upon today!"
- "Oh no! Bad luck old chum! Better luck next time!"
- "Hmmm, some coin not too shabby for the other vendors around."
- "Wanna give it a shot? You can only win if you know how to spin!"
- "Try again? The wheel may just be on your side!"
- (Player to NPC) "Lets Spin it Right Round Baby!"
- (Player to NPC) "I'm not feeling so lucky right now."

High Concept Statement:

Dungeon Festival is a rogue-like dungeon crawler game where the player discovers a secret forest with a dungeon entrance. The player soon realizes that the legends of the yearly dungeon festival is real and enters to meet different spirits and adventures. The player is the only human in this festival and has to attend the sport of Dungeoneering to enhance their fame and experience. Through a series of diverse dungeon terrains, the player will have to rely on their skill in battle to enhance their weapons and skill tree to prove their worth in the festival.

Genre:

Rogue-like Dungeon Crawler

Story Summary:

The human character will discover a forest with a hidden cave entrance. Inside the cave are NPC's who are not human and will inform the human character that they must attend the sport of dungeoneering in order to enhance their fame and experience in their cave.

Target/Demographic:

- **Gender:** All
- **Ages:** 12-17+
- **Rated:** T for teen
- **Location:** USA
- **Language:** English (Spanish, French (common))
- **Price:** \$5-\$10
- **Customer Interest:** Rogue-like, Dungeon Crawlers, Team-building games, RNG generation, Crafting, Unique Art,

Jira

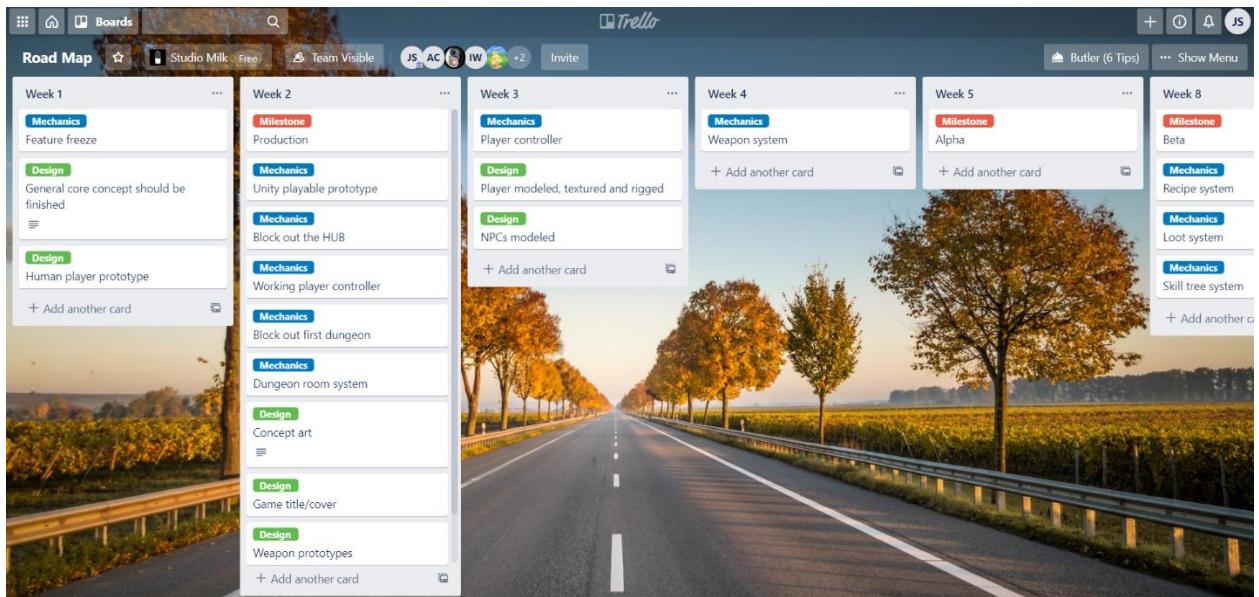
Projects / Studio Milk / SMJ board

SMJ Sprint Q2 Week 7

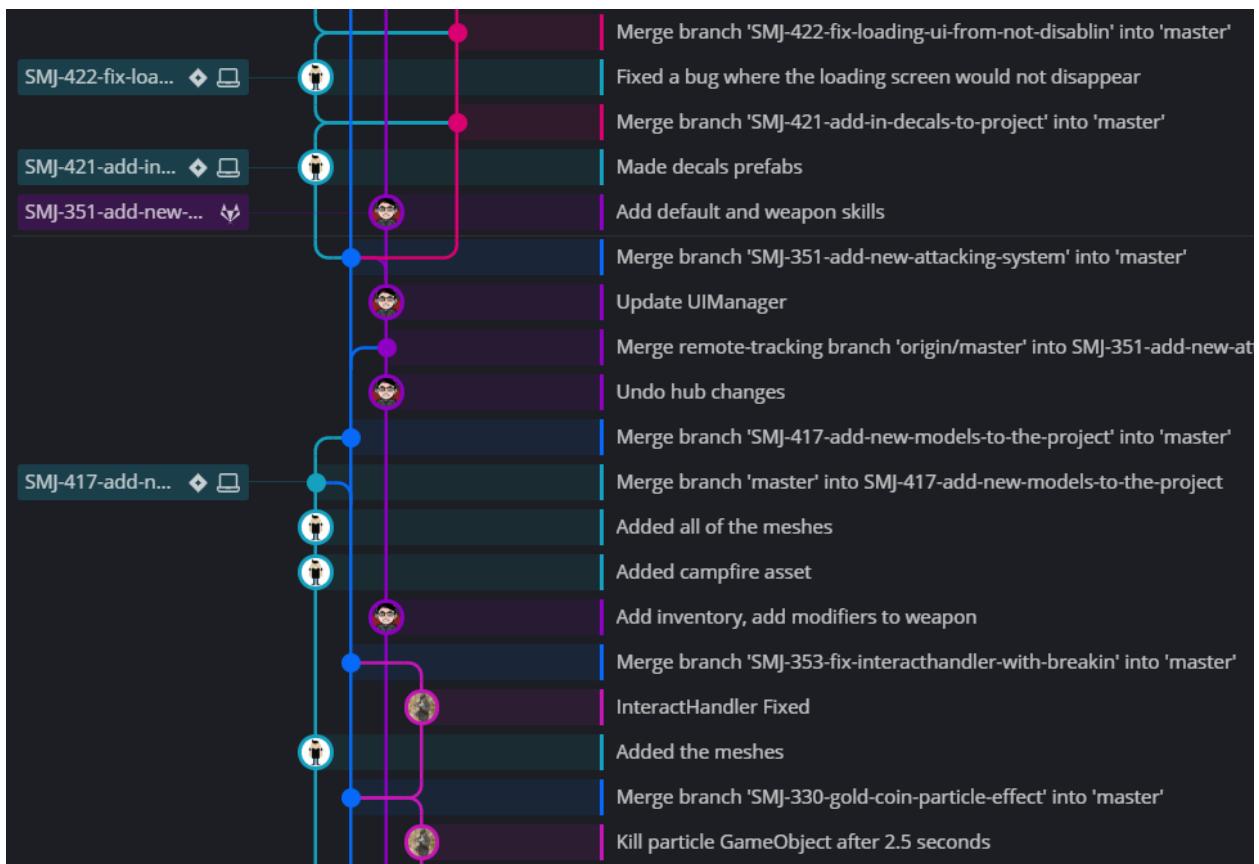
Only My Issues Recently Updated

TO DO	IN PROGRESS	DONE
Animate Vine Lady with all animations SMJ-397	Create a data/player save system Beta 2.0 SMJ-347	Fix braking crates. Too much gold and pieces Beta 2.0 SMJ-354
Add new animations to the player rig Alpha 3.0 SMJ-196	Allow players to join other games Alpha 3.0 SMJ-276	Re-Program Consumable Usage with input system Beta 2.0 SMJ-452
Create a loading wheel sprite SMJ-90	Fix the camera when a crate is broken SMJ-414	Make lighting better SMJ-438
Create game banner (1920x1080 & 1375x2125) SMJ-11	Texture candelabra Beta 2.0 SMJ-378	Create modifier list for Kyle SMJ-444
Change bush color Beta 1.0 SMJ-277	Texture DungeonFloorGrate Beta 2.0	Cleanup console SMJ-451
Texture wagon		Add another iteration for the hub area

Trello



GitKraken



Flow Charts & Diagrams

Gameplay and player progression diagram

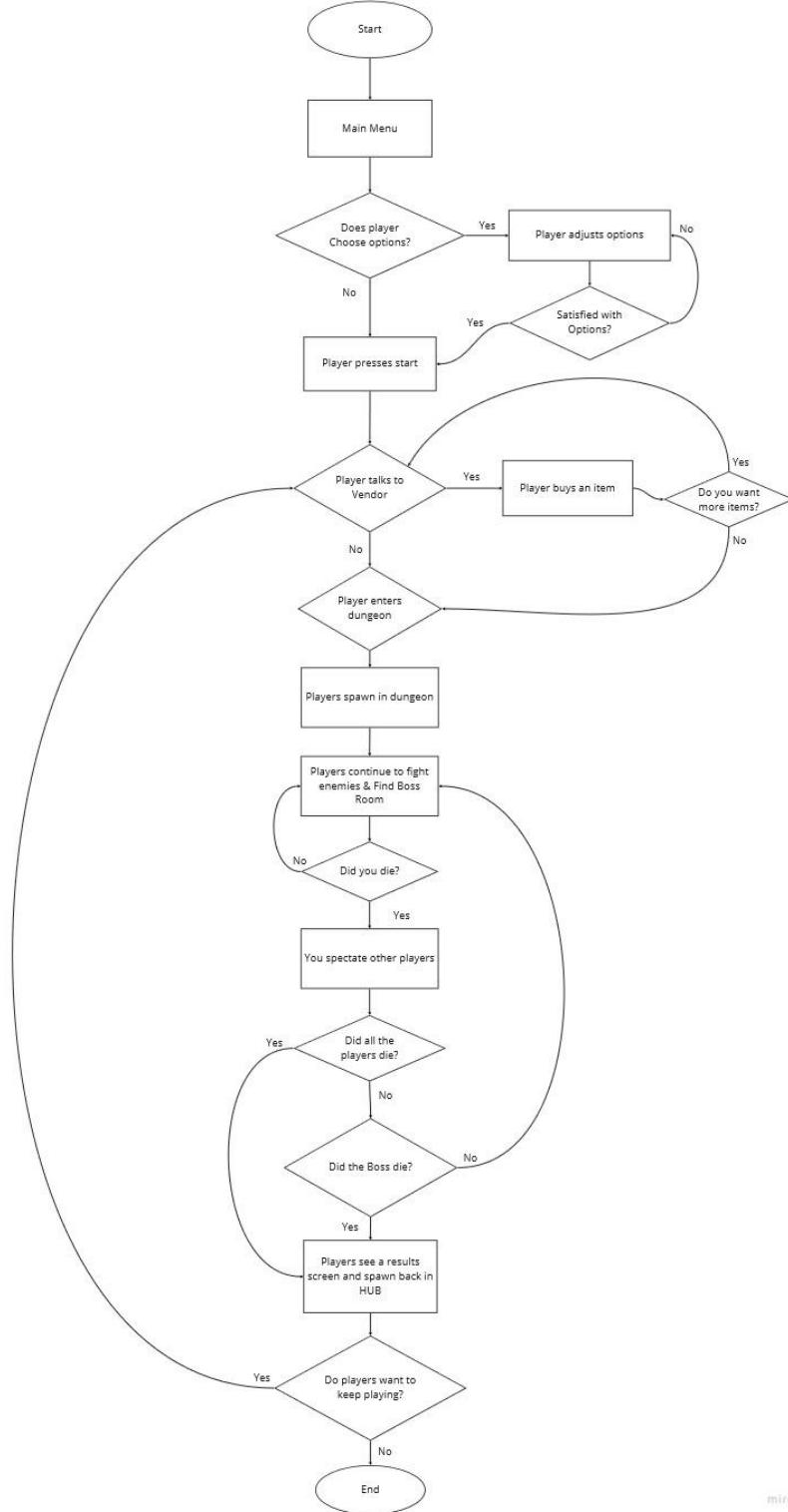


Diagram explaining how the dungeon is generated

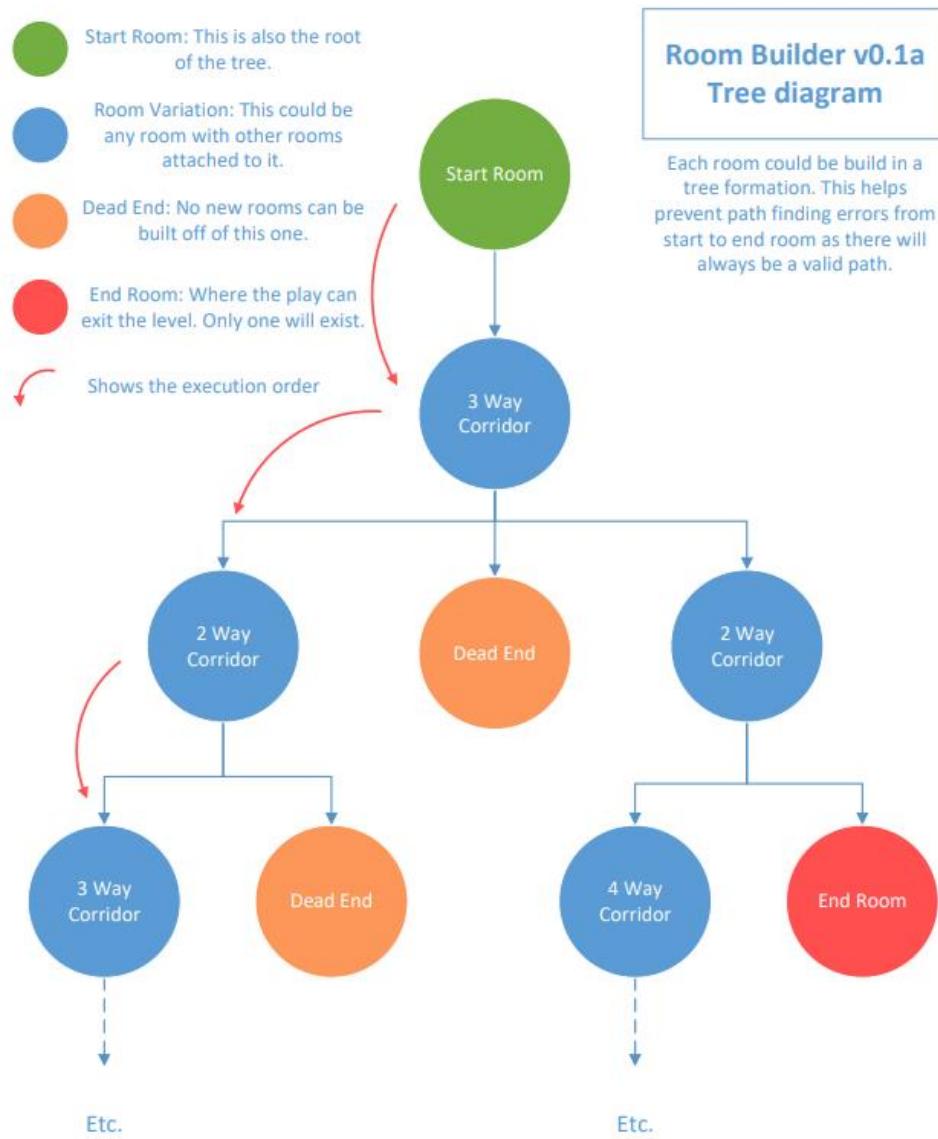
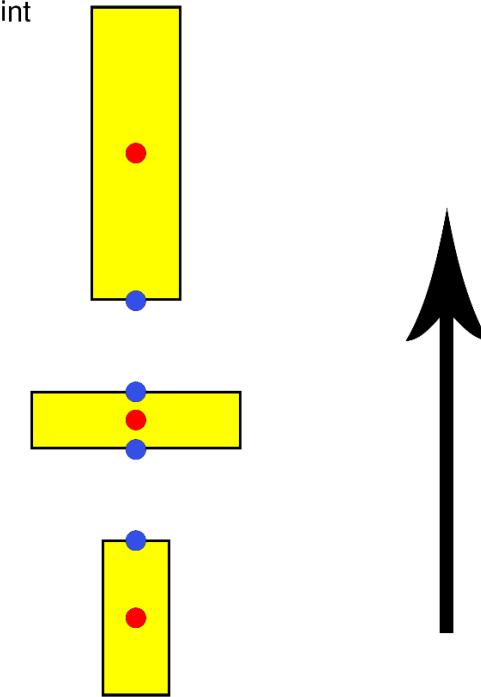


Diagram showing the designers how the swords will be dynamically built in game.

- Pivot Point
- Connect Point



Concept Art

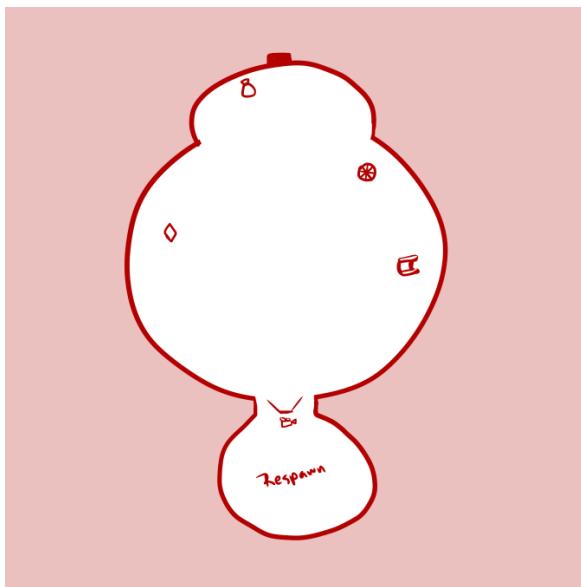
Early & Alpha Concept Art

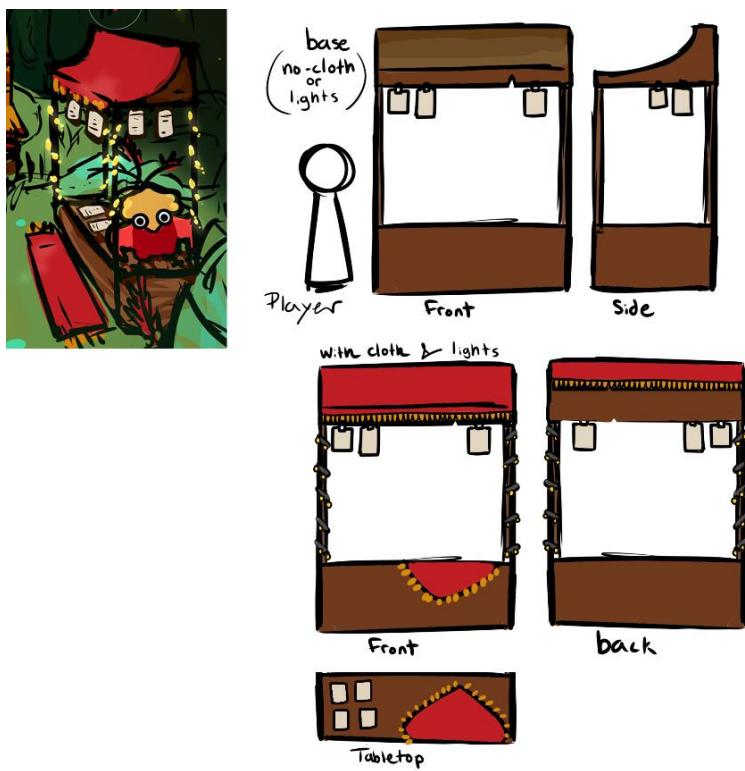
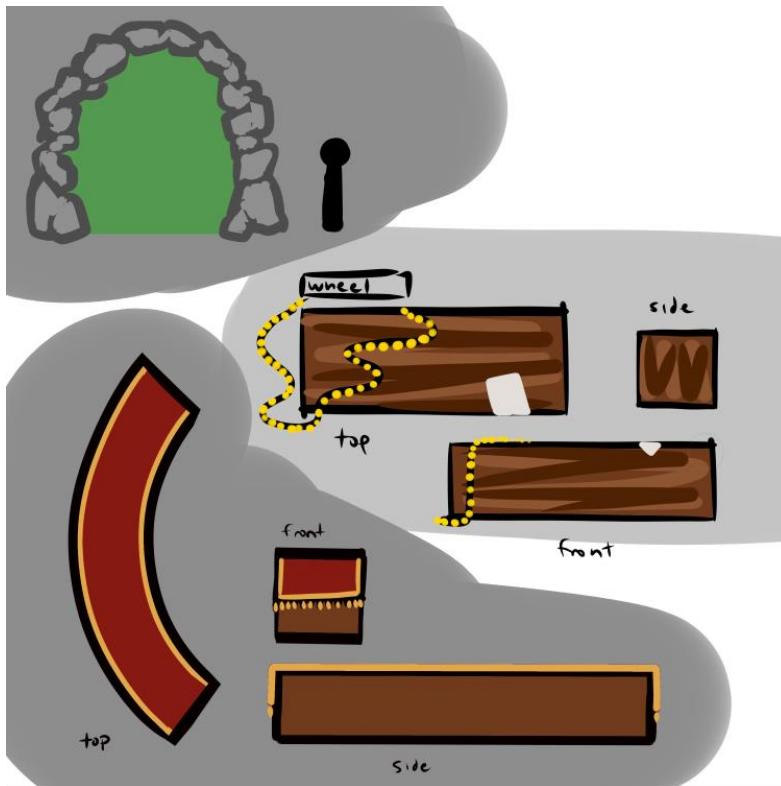
These pieces of art was created before the team started developing the game. This become the foundation and inspiration in the creation of Mythgrove.





The Hub





Player



NPS's

Early image of all of the possible NPS's. The team's top four was chosen.

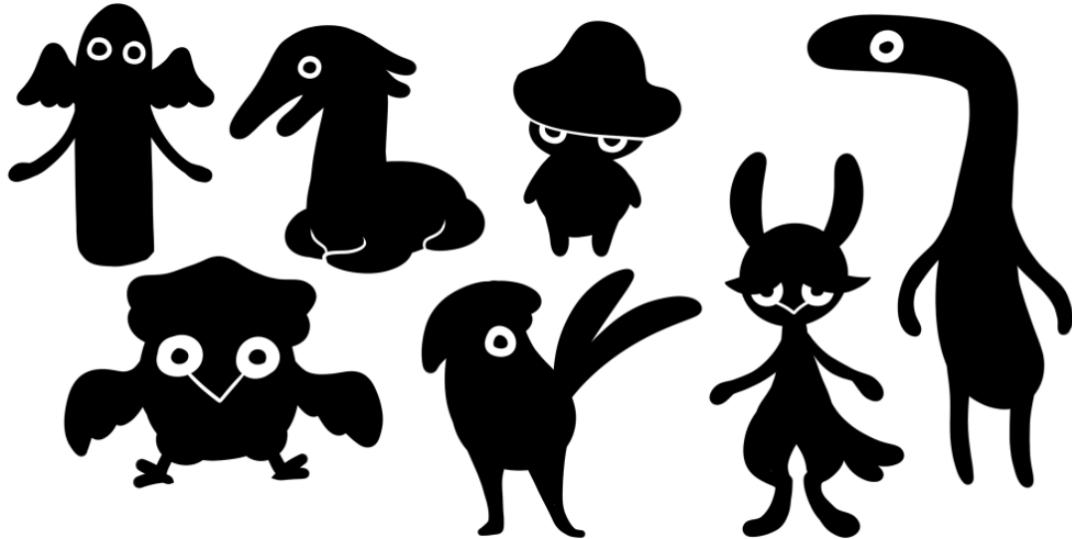
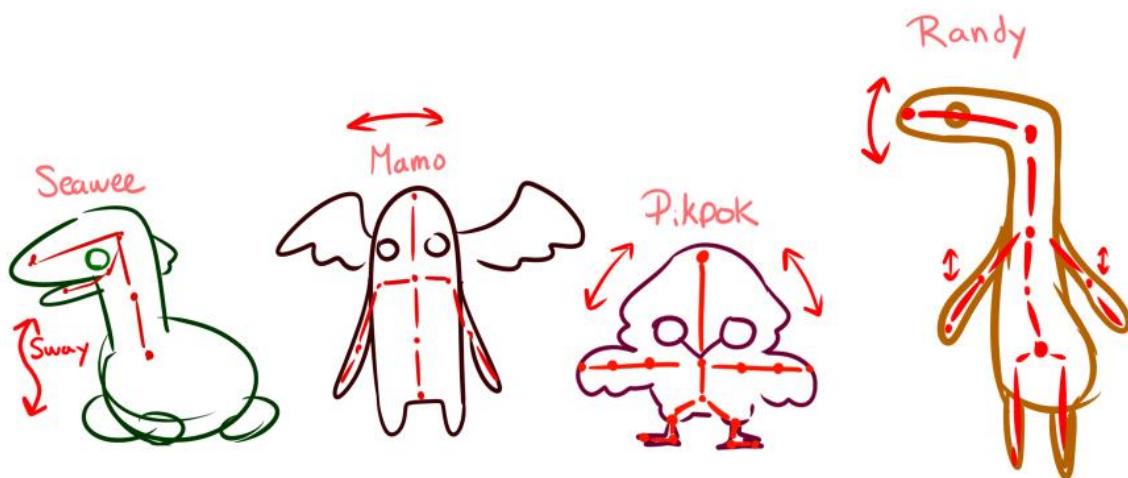
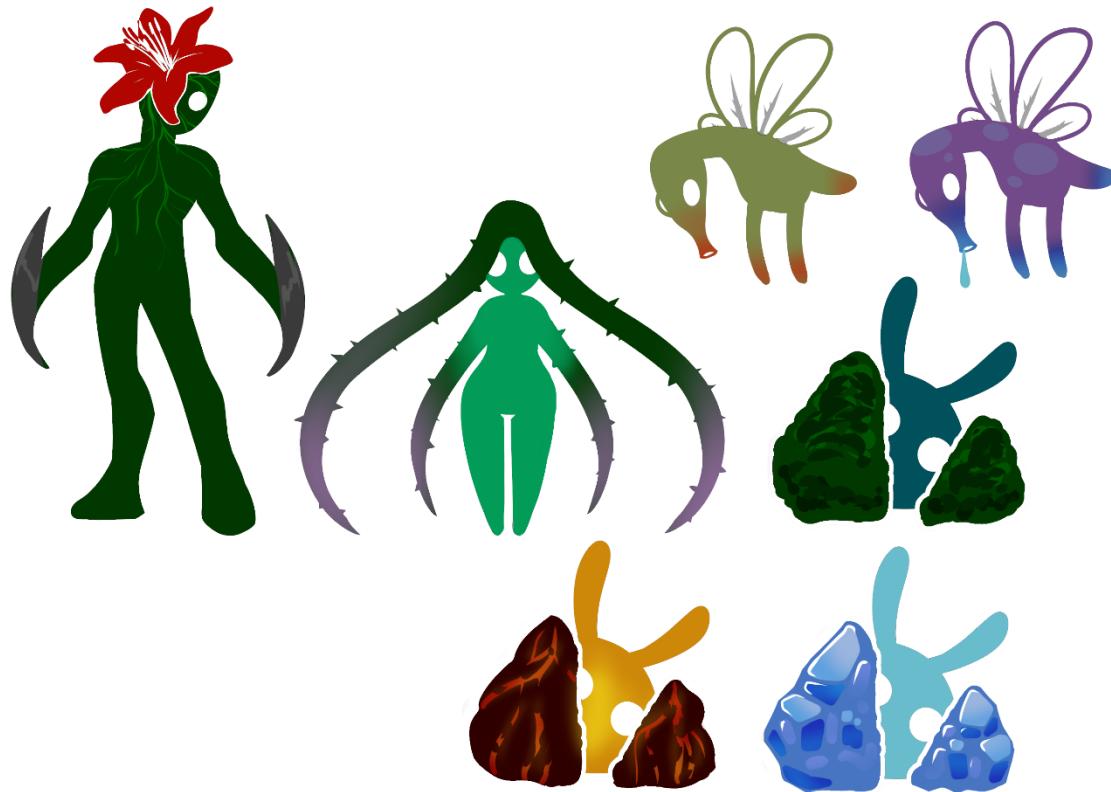


Image showing the winning choices. Names were also given.



Monsters



Props



Color Palette



Fire Brick
#BE2026



Dark Olive
Green
#413E1B



Dark Sea
Green
#79AA60



Dark Khaki
#B2C172



Dark Olive
Green
#707042

Reasons for the Colors:

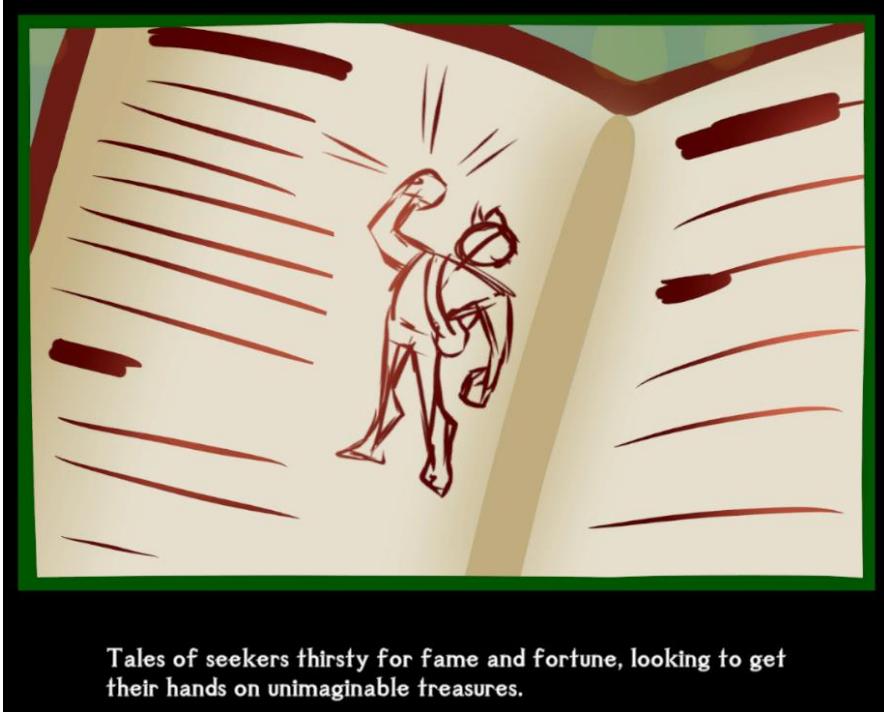
Red/Green is a complimentary palette, and we wanted to keep the natural feeling of the greens, but the vibrancy of their reds as well. Red/Gold is a common symbolic combination in China, for celebrations and festivals, and so that's why the main colors ended up being green, red, and gold.

Opening Cinematic

Prototype



The legend had been passed down for years.



Tales of seekers thirsty for fame and fortune, looking to get
their hands on unimaginable treasures.



Some that find the portal come home to tell the tale



but those that enter never return.



We do not know what lies beyond that light in the woods,



but we can only hope that one day a brave adventurer will
step back through the portal,



and reveal to us the mysteries of MythGrove.

Final





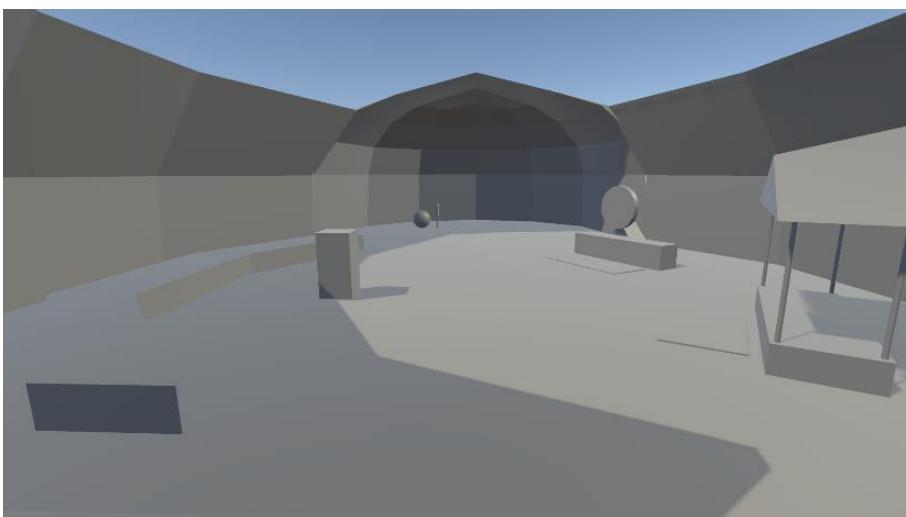
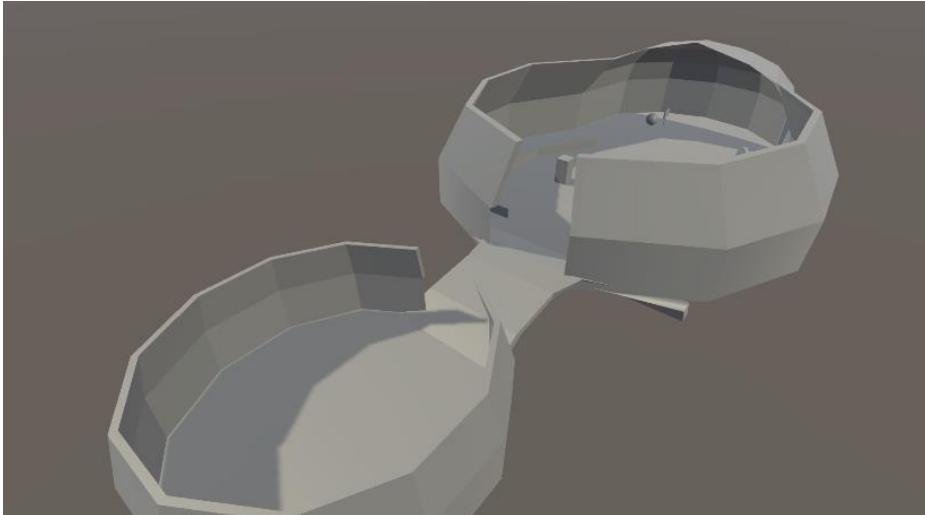




Game Stages

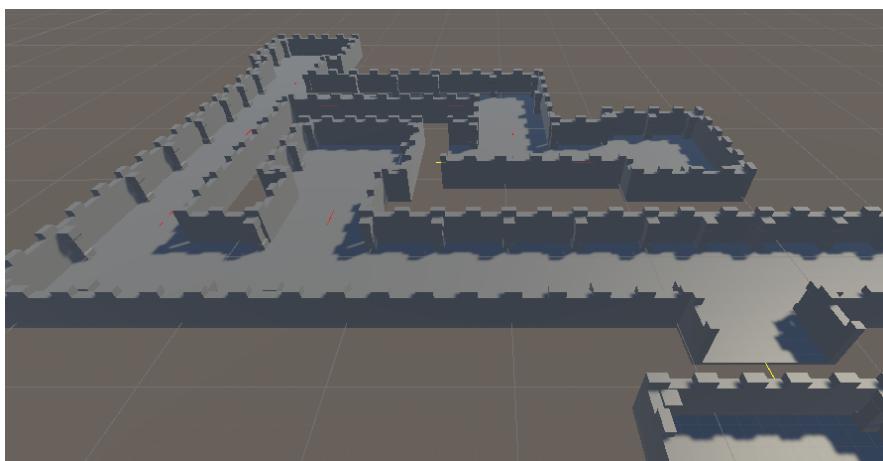
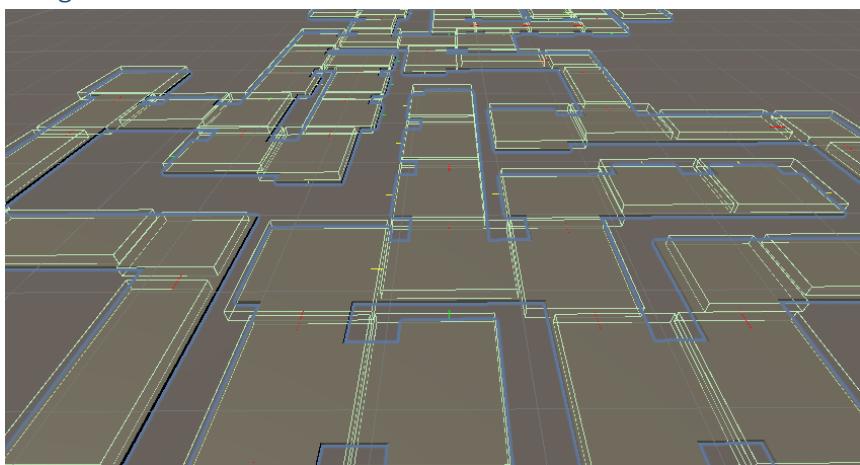
Prototype

Hub



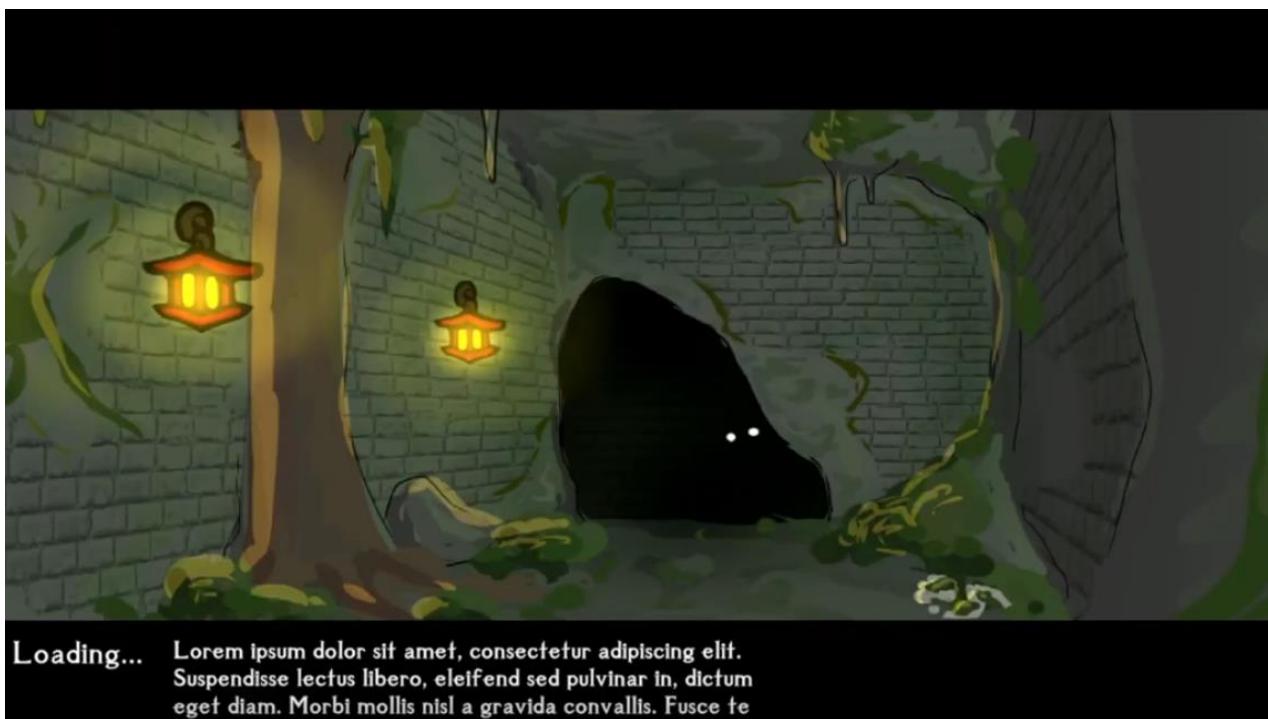


Dungeon

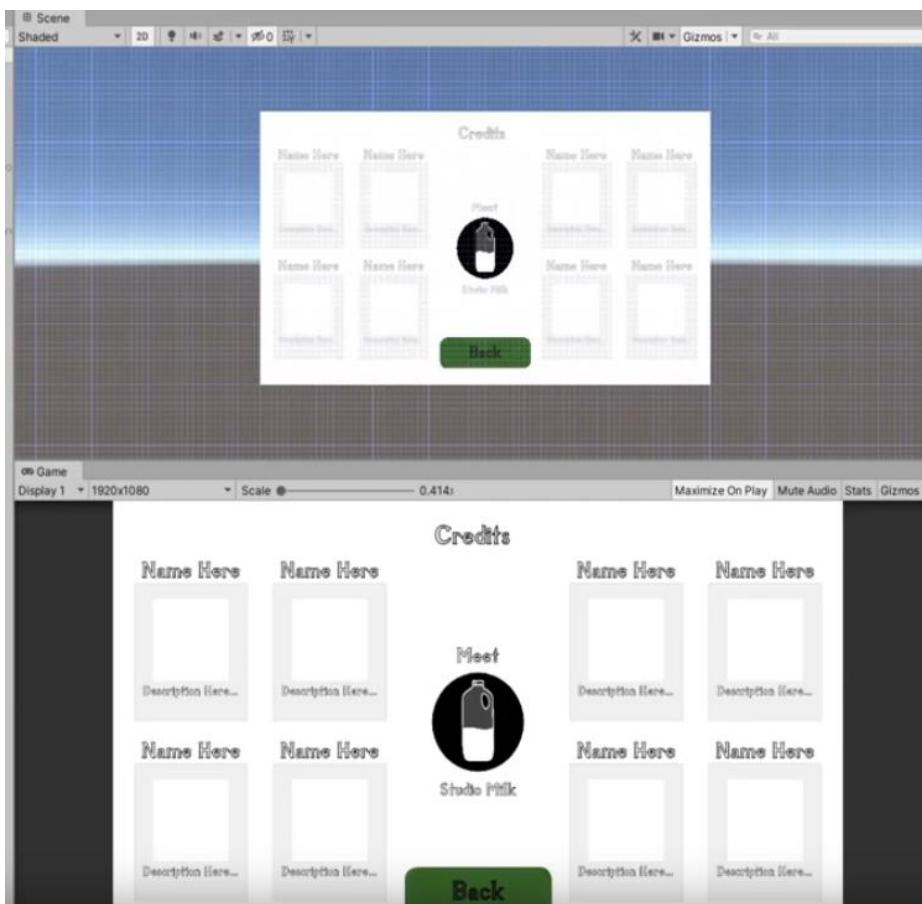


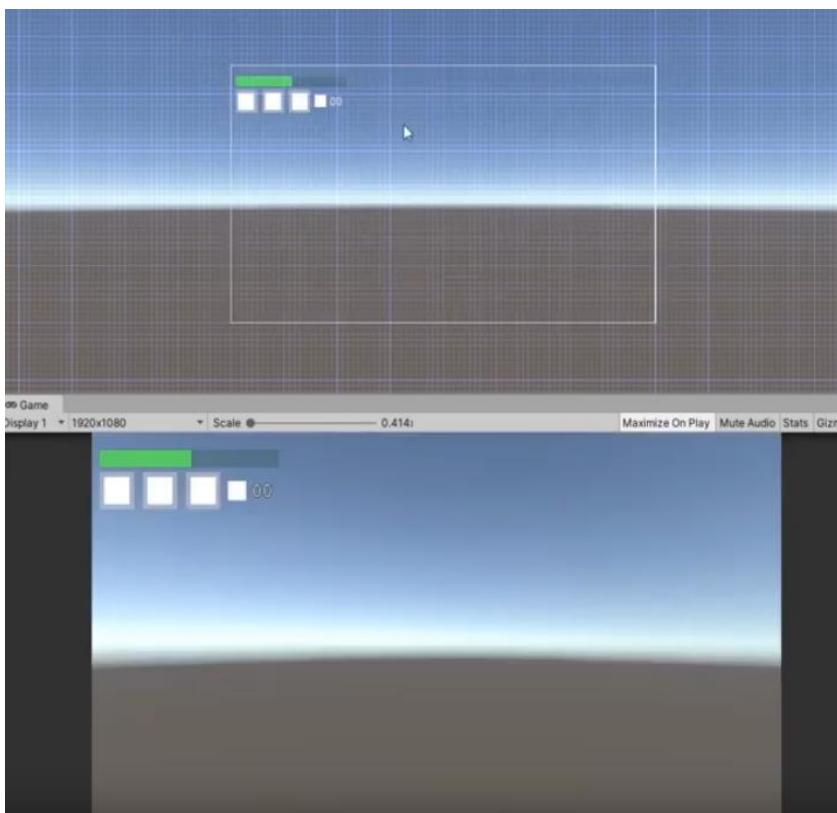
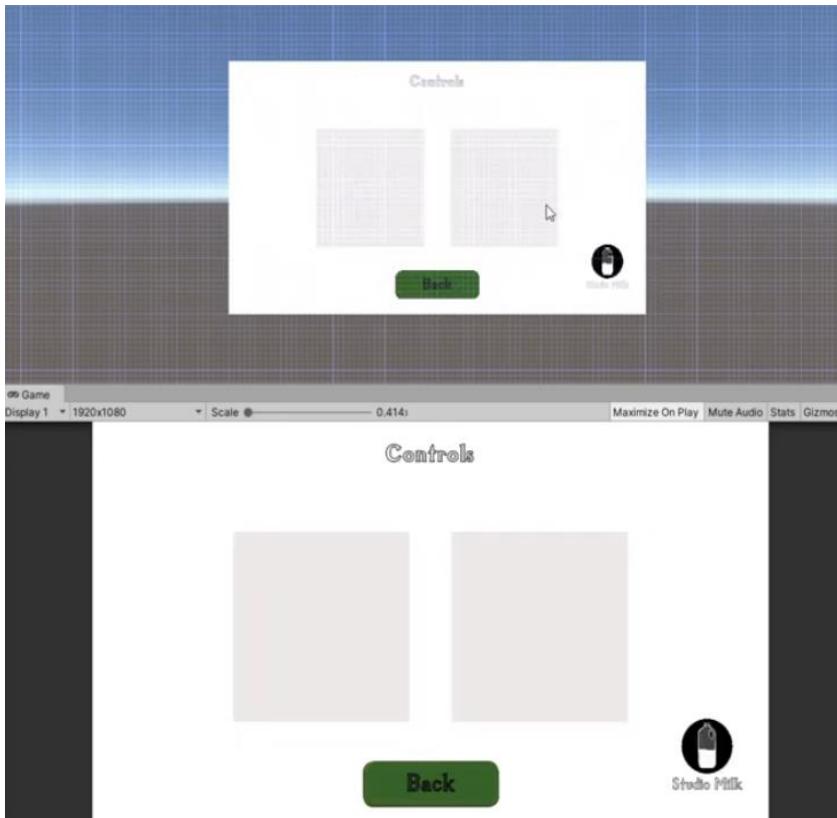


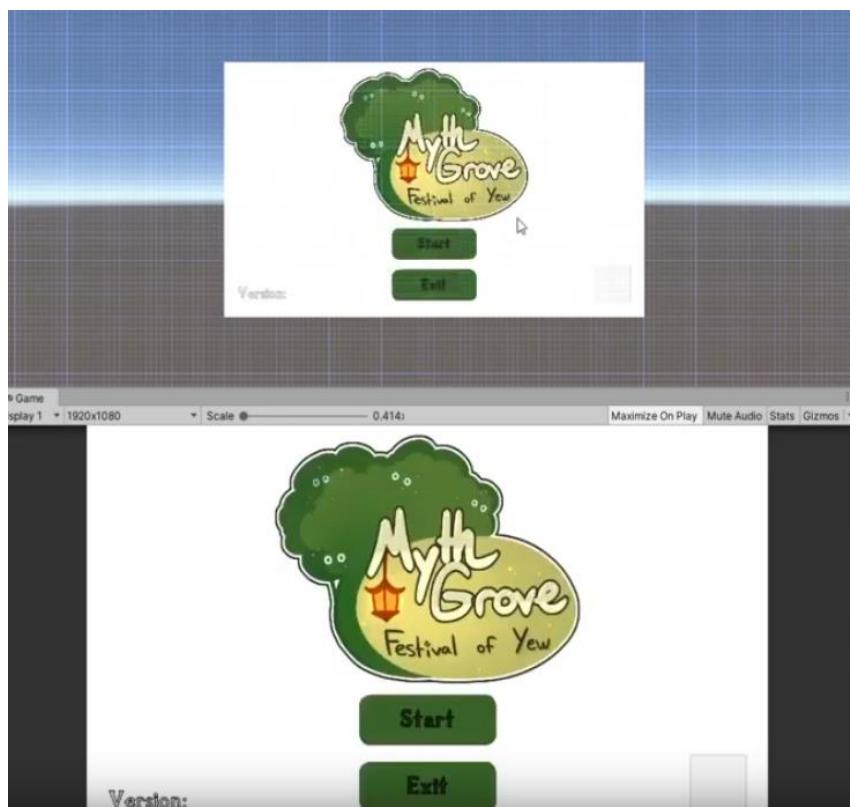
UI

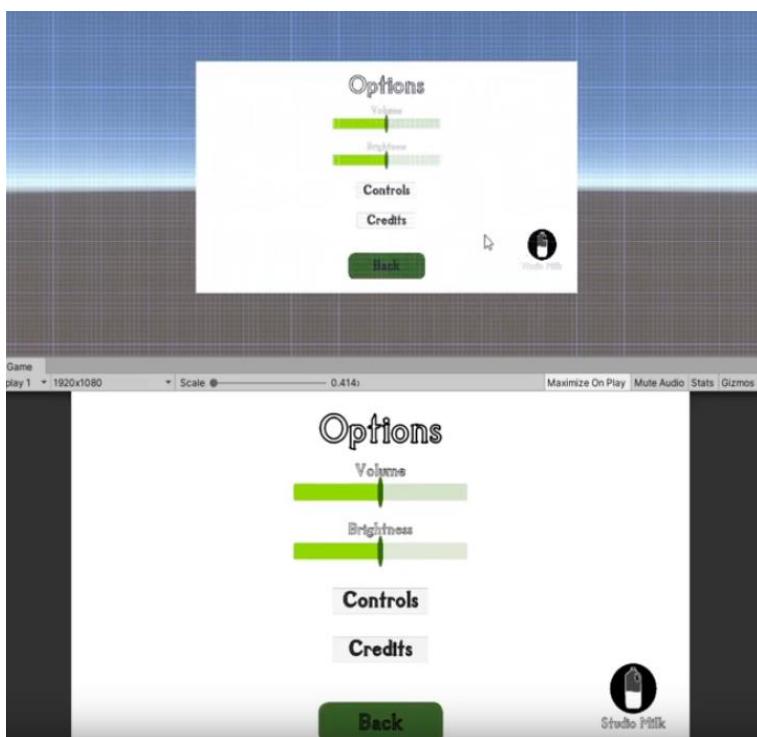
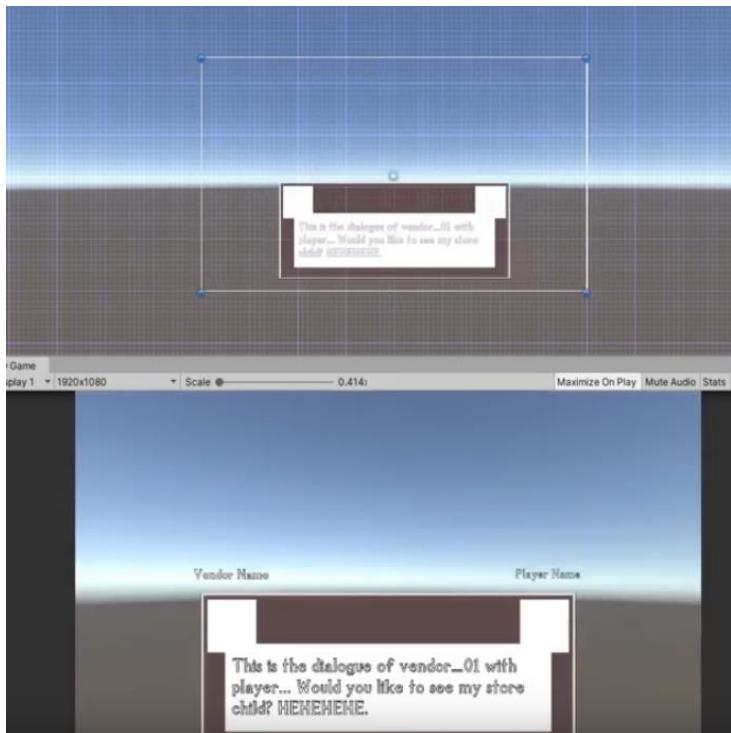


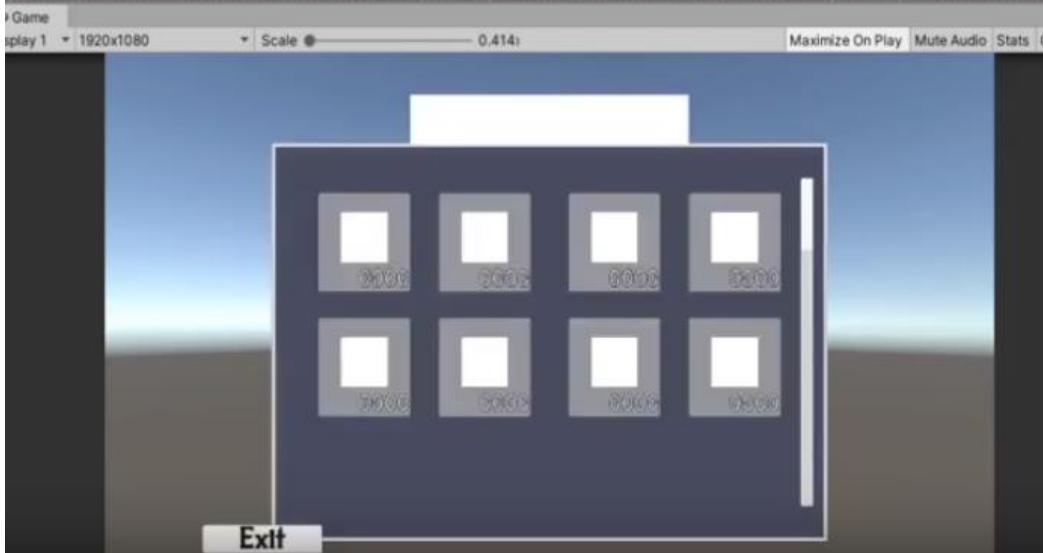
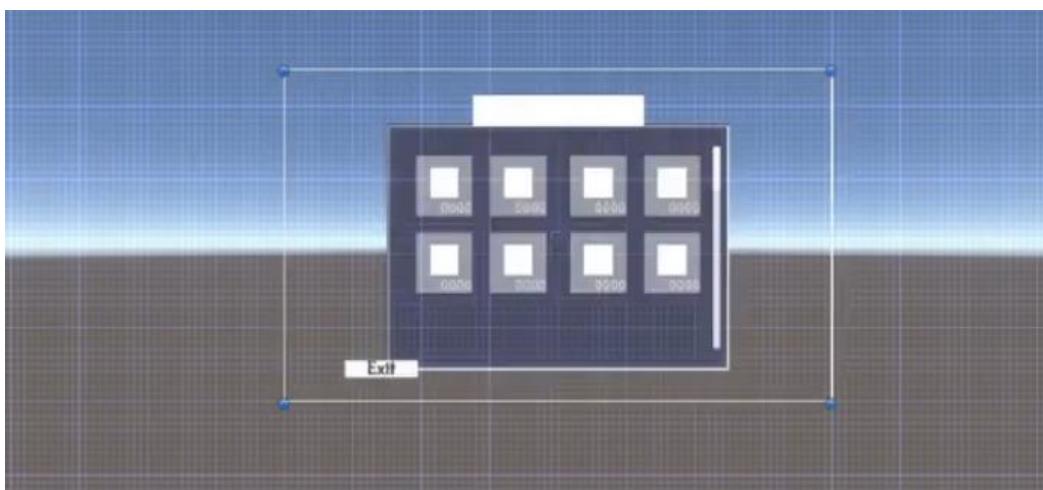
Loading... *Lorem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse lectus libero, eleifend sed pulvinar in, dictum eget diam. Morbi mollis nisl a gravida convallis. Fusce te*

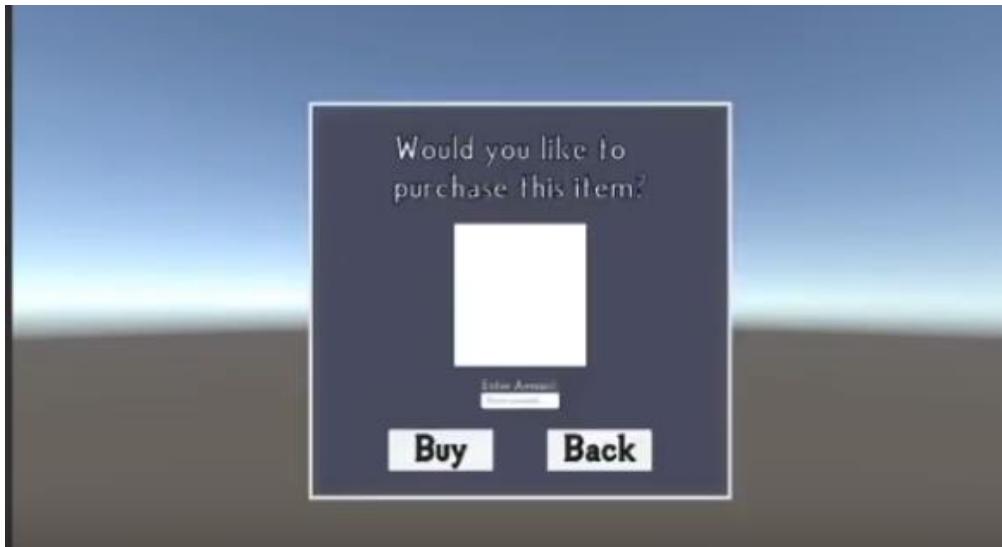












Alpha

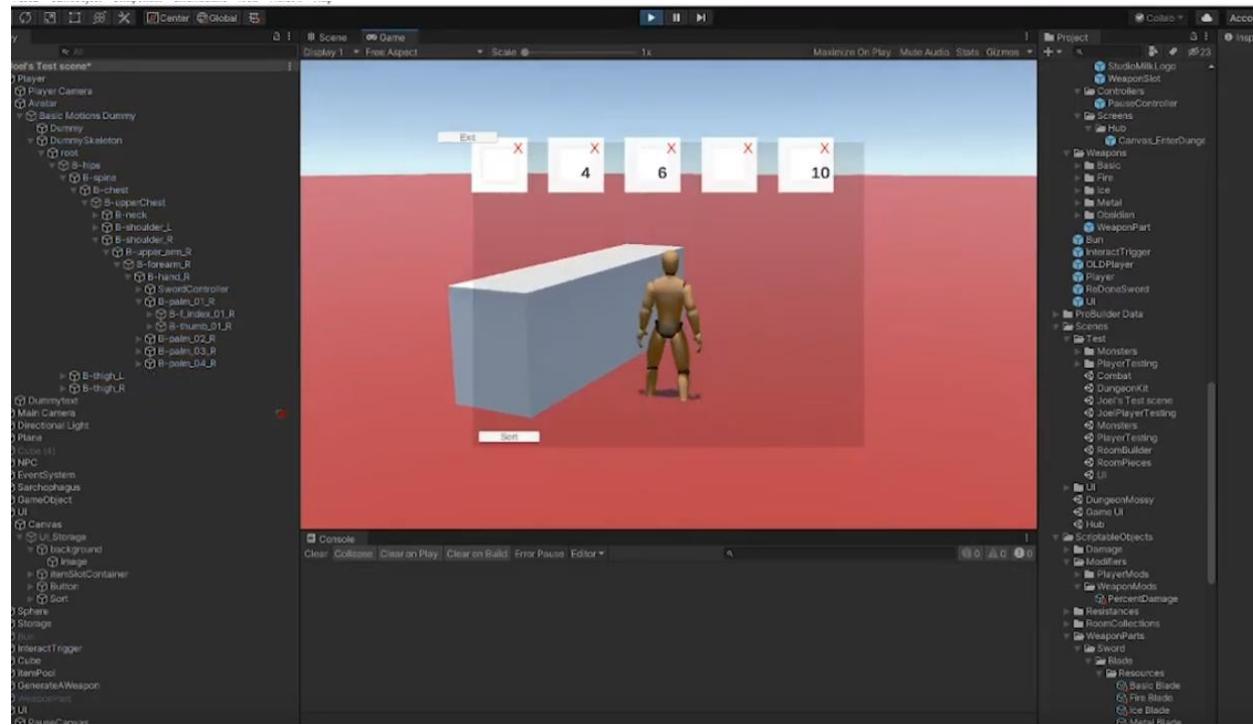
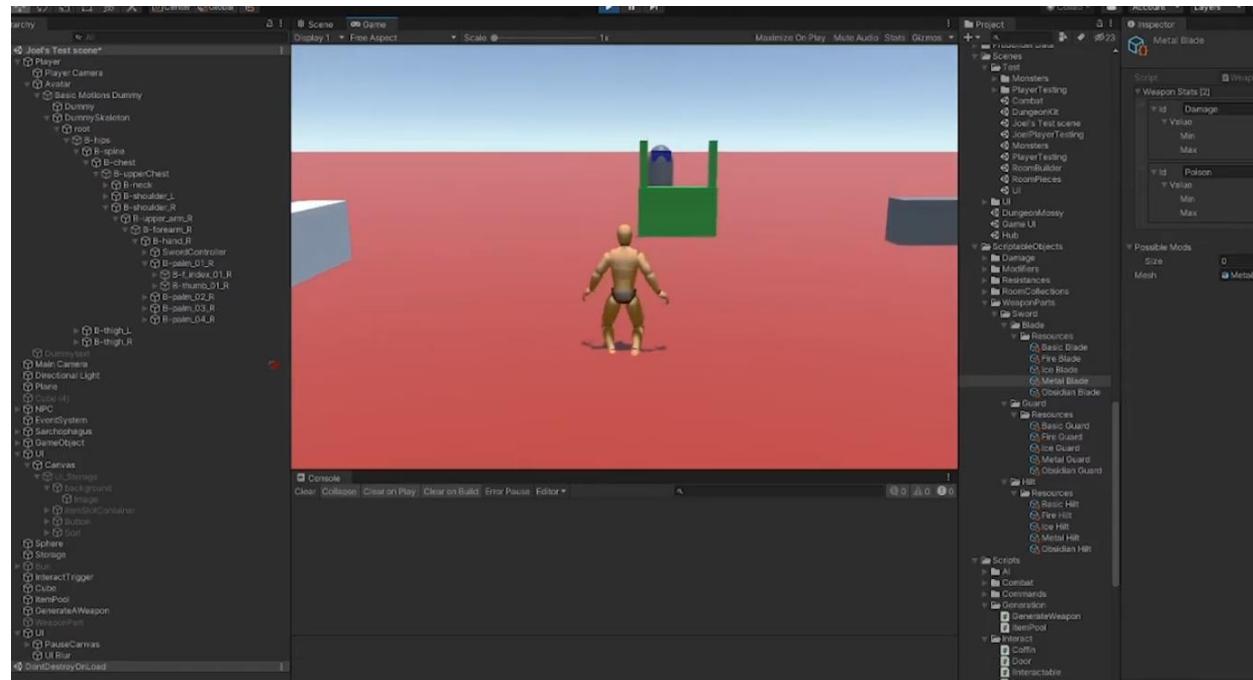
Hub

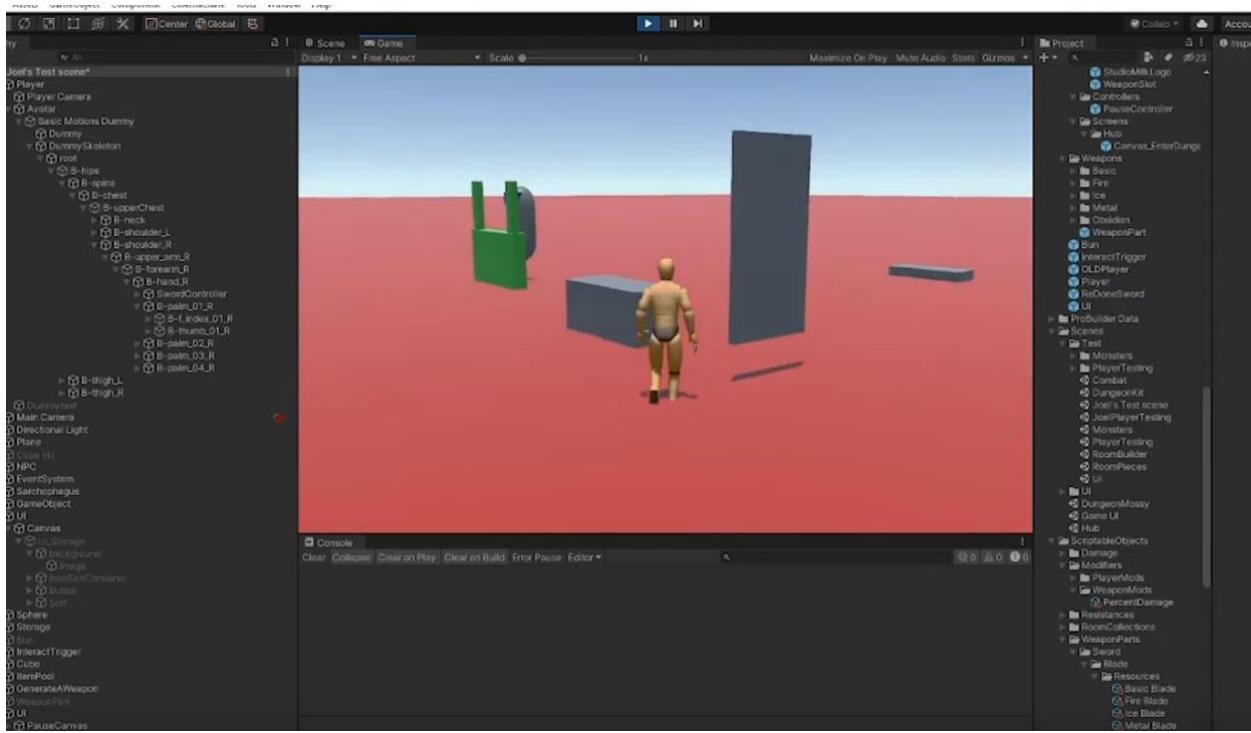






Shops





Dungeon

Iteration 1



Iteration 2







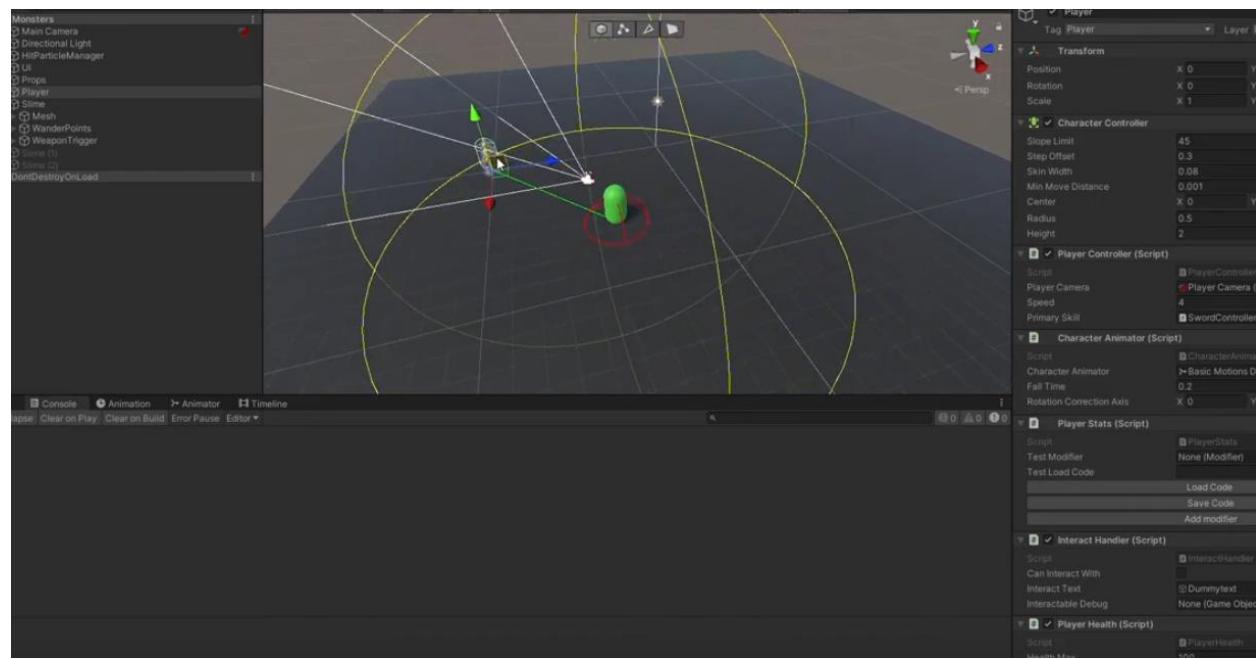






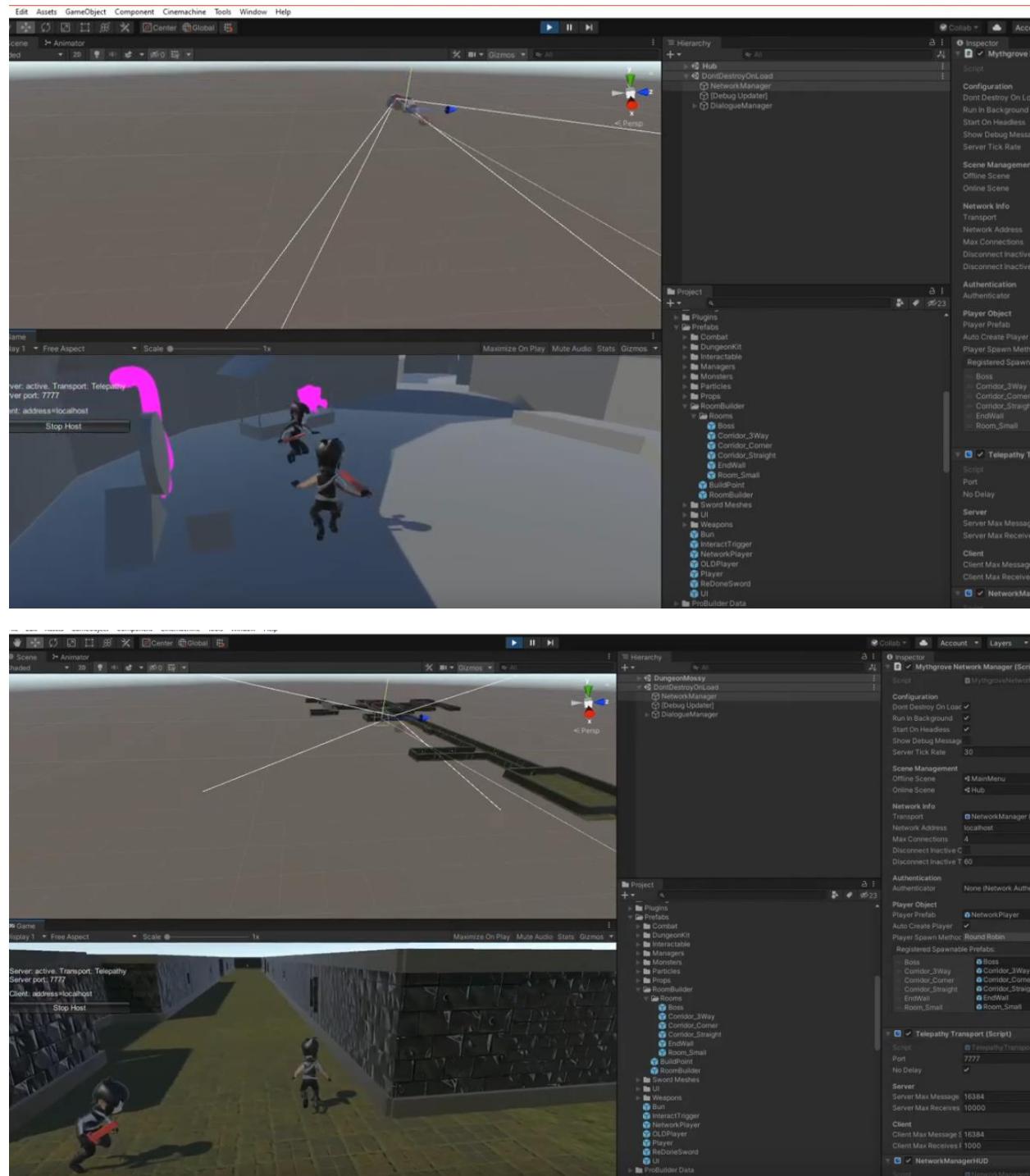


Combat





Networking



UI



Beta

Hub











Dungeon



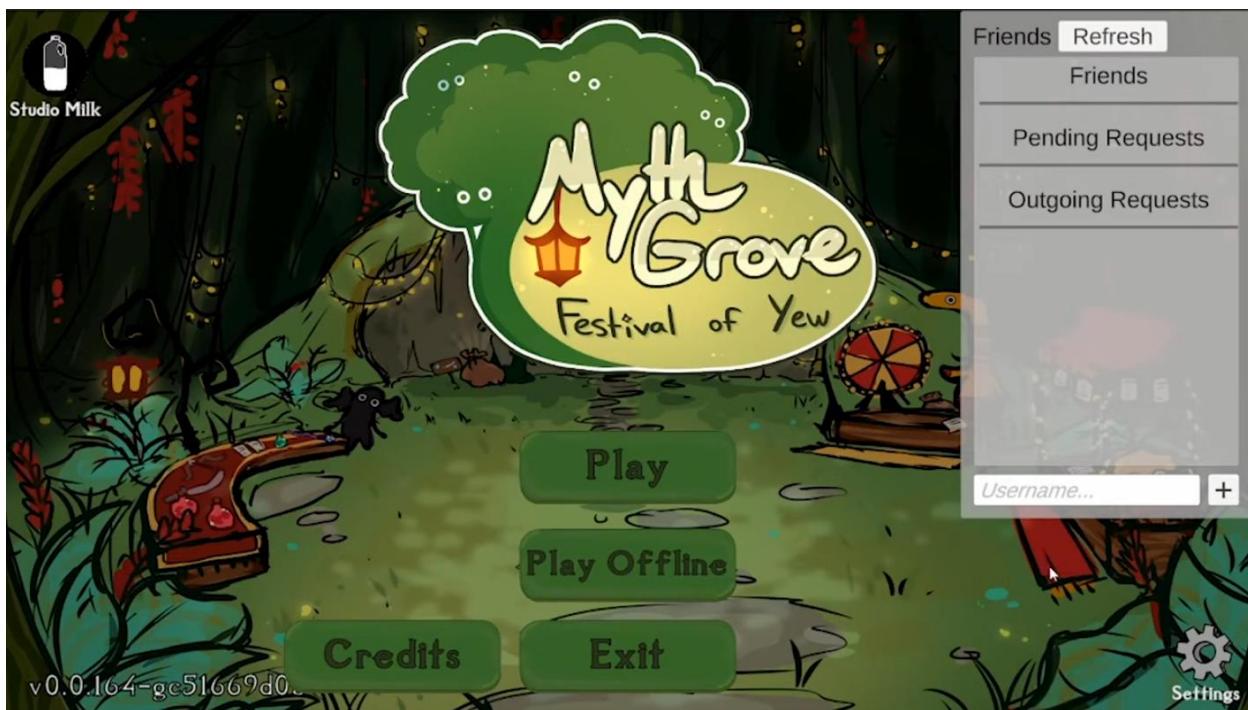


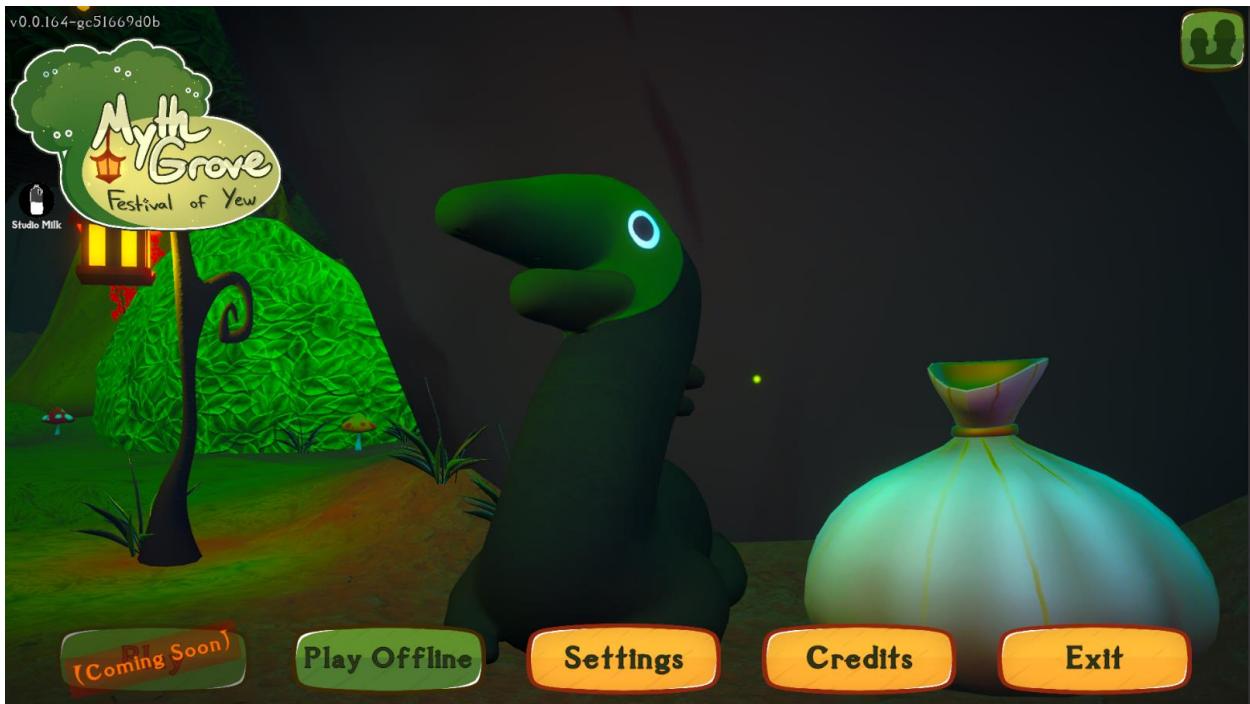






UI







Release**Hub**











Dungeon

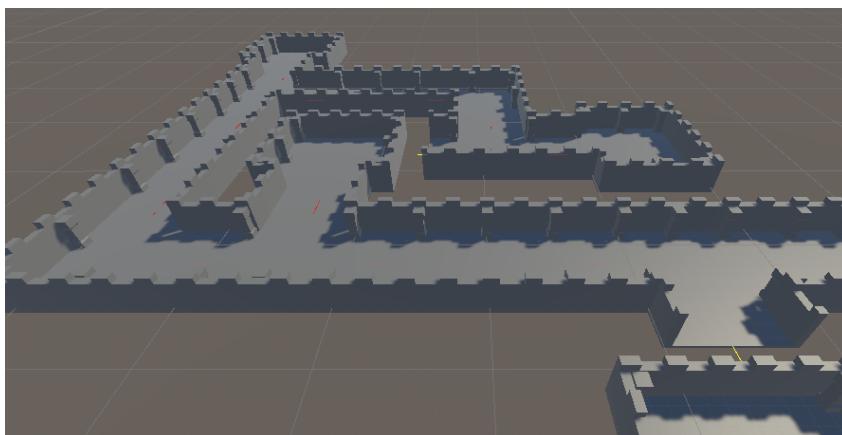
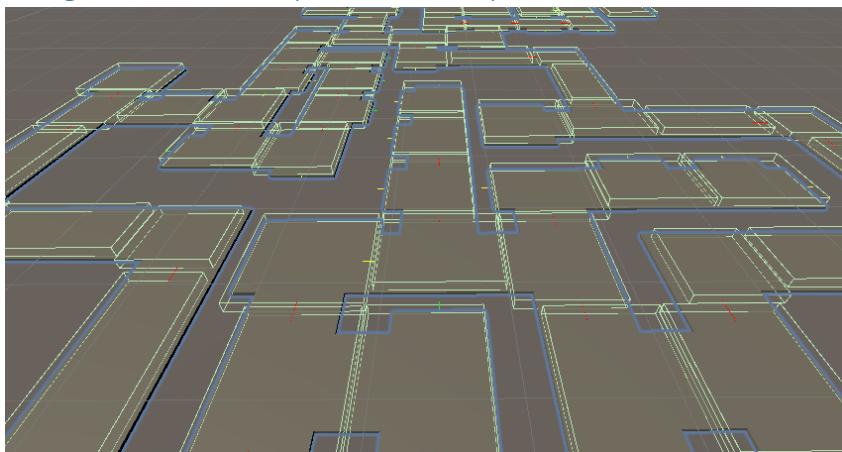


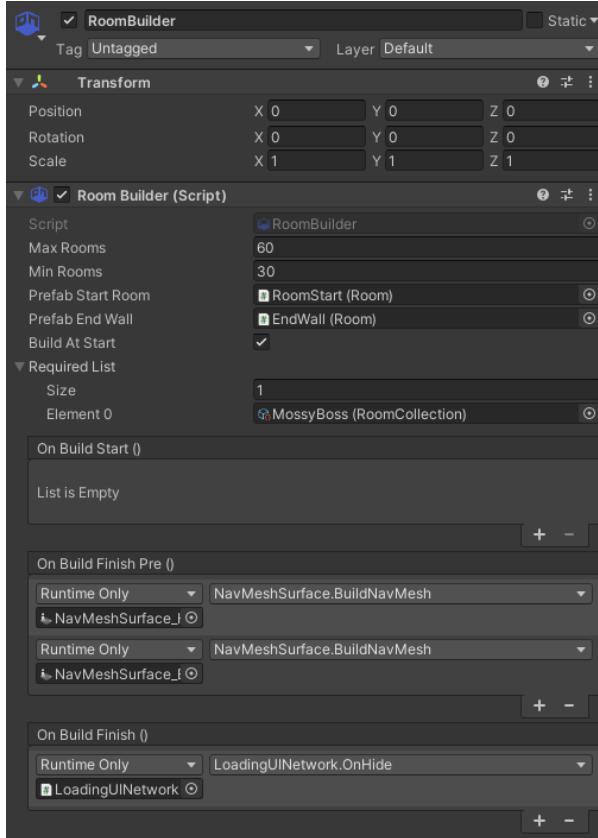




Mechanics

Dungeon Generator (Room Builder)





Start Method

```

///<summary>
/// Main method for starting the build process
///</summary>
[ContextMenu(itemName: "Build")]
↳ Frequently called 3 usages  Jason +2
public void StartBuildingLevel() {
    //Setup the first room
    roomStart = Instantiate(prefabStartRoom, Vector3.zero, Quaternion.identity, transform);

    if(NetworkServer.active)
        NetworkServer.Spawn(roomStart.gameObject);

    listRoomColliders.AddRange(collection: roomStart.colliderGroup.GetComponentsInChildren<Collider>());

    onBuildStart?.Invoke();

    //Start the building process
    StartCoroutine(routine: BuildRoom(roomStart));
}

```

Choose Room Method

```
/// <summary>
/// Chooses a random room from the list of room attachments. Accounts for the minimum room requirement.
/// </summary>
/// <param name="baseRoom"></param>
/// <returns></returns>
/// <exception cref="MissingRoomAttachmentsException"></exception>
↳ Frequently called  ↳ 1 usage  ↳ Jason
private Room ChooseRoom(Room baseRoom) {
    //Are there no room attachments?
    if(baseRoom.roomAttachments.Count <= 0)
        throw new MissingRoomAttachmentsException();

    //Loop through all of the possible rooms that can be created from the current room
    for(int i = 0; i < baseRoom.roomAttachments.Count; i++) {
        RoomAttachment roomAttachment = baseRoom.roomAttachments[i];
        RoomCollection roomCollection = roomAttachment.roomCollection;
        Room currentRoom = roomCollection.GetRandomRoom();

        //Check if the room has a minimum
        if(roomAttachment.useMin) {
            //Check for an existing key
            if(dictionaryMinRooms.ContainsKey(baseRoom) && dictionaryMinRooms[baseRoom].ContainsKey(currentRoom)) {
                int amount = dictionaryMinRooms[baseRoom][currentRoom];

                if(amount < roomAttachment.min) {
                    dictionaryMinRooms[baseRoom][currentRoom] = dictionaryMinRooms[baseRoom][currentRoom]++;
                    return currentRoom;
                }
                //If the amount is greater than or equal too
                continue;
            }
            //Create the key
            if(!dictionaryMinRooms.ContainsKey(baseRoom))
                dictionaryMinRooms.Add(baseRoom, new Dictionary<Room, int>());
            dictionaryMinRooms[baseRoom].Add(currentRoom, 1);
            return currentRoom;
        }
    }
    return GetRandomRoom(baseRoom);
}
```

Recursion Method

```
//Pick the room that we will be building
Room pickedRoom = ChooseRoom(currentRoom);

//Build the room at the chosen build point
Room newRoom = BuildRoomAtPoint(pickedRoom, buildPointStart);

//Did the build process fail
if(newRoom == null) {
    buildPointStart.BuildPointState = BuildPointState.DeadEnd;
    threadCounter = 0;
    continue;
}

//Add all of the colliders from the new room
listRoomColliders.AddRange(collection: newRoom.colliderGroup.GetComponentInChildren<Collider>());

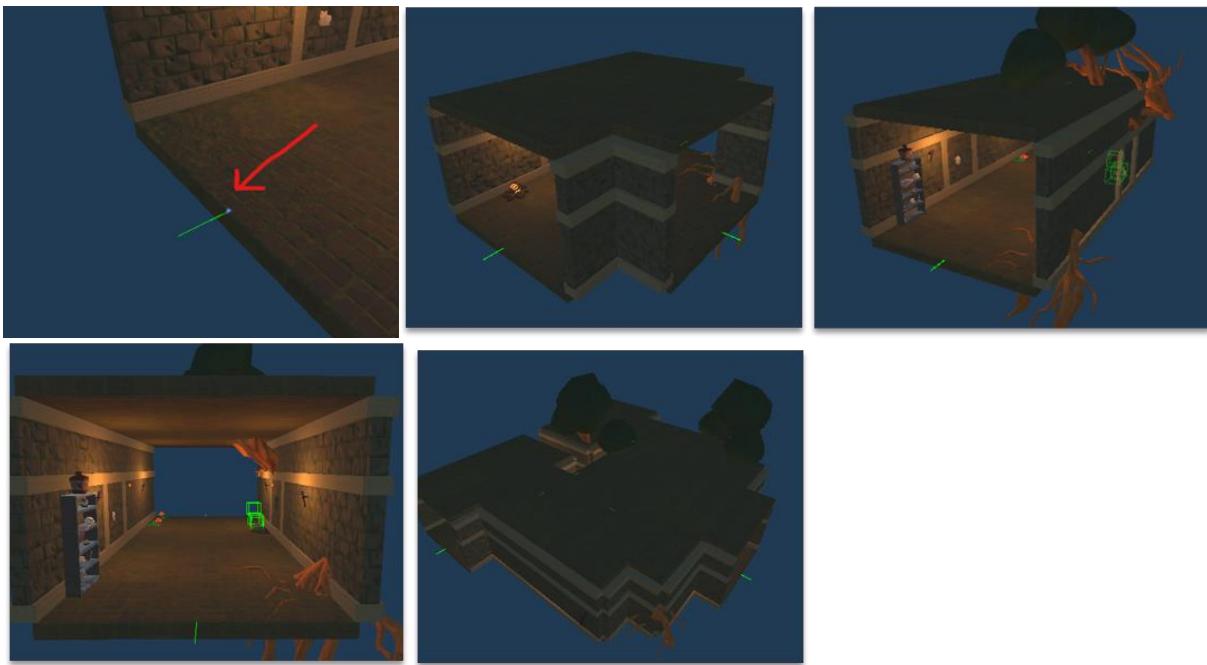
//Store the room
listRooms.Add(newRoom);

//Debug: Slow mode
if(slowMode)
    yield return new WaitForSeconds(1);

threadCounter++;

//Recursion
roomsBuilt++;
yield return StartCoroutine(routine: BuildRoom(newRoom));
```

Rooms



Monsters

```
namespace Mythgrove.AI {
    ⚡ No asset usages   ⚡ 18 usages   ⚡ 2 inheritors   ⚡ JasonSkillman   ⚡ 2 exposing APIs
    public abstract class Monster : NetworkBehaviour, IAttackable {

        [Header("Monster")]
        public float maxHealth = 100; ⚡ 0+ changes
        public float attackRadius = 3.0f; ⚡ 0+ changes
        public DamageData damageData; ⚡ 0+ changes
        public ResistanceData resistanceData; ⚡ 0+ changes

        [Header("References")]
        public HealthBarUI healthBar; ⚡ 0+ changes

        [Header("Debug")]
        public bool canAttack = true; ⚡ 0+ changes
        public bool isInvincible; ⚡ 0+ changes
    }
}
```

Some networking code from the monster class

```
#region Attacking

/// <summary>
/// Attacking the player
/// </summary>
/// <param name="obj"></param>
/// <param name="attackable"></param>
[Server]
protected virtual void DamageTriggerOnTrigger(GameObject obj, IAttackable attackable) {
    //Can the monster attack?
    if(!canAttack) return;

    //Send damage to the player
    Player player = obj.GetComponent<Player>();
    player.OnAttacked(sender: this, damageData.damage, damageData.damageType);
}

/// <summary>
/// Callback after sending the damage to the player. Player already took damage
/// </summary>
/// <param name="actualDamage"></param>
[Server]
public virtual void DamageTriggerCallback(float actualDamage) {
    //Cap actualDamage to 0
    if(actualDamage < 0) actualDamage = 0;
}
```

```
#region Attacked

[0+3 usages 2 JasonSkillman]
public virtual float OnAttacked(float damage, DamageType damageType = DamageType.Raw) {
    //Apply resistance
    float damageTaken = resistanceData.ApplyResistance(damage, damageType);

    //Send data to the server
    CmdOnAttacked(damageTaken, damageType);

    //Return data back to the sender
    return damageTaken;
}

[Command(ignoreAuthority = true)]
[1 usage 2 JasonSkillman]
public virtual void CmdOnAttacked(float damageTaken, DamageType damageType, NetworkConnectionT
    if(IsDead) return;

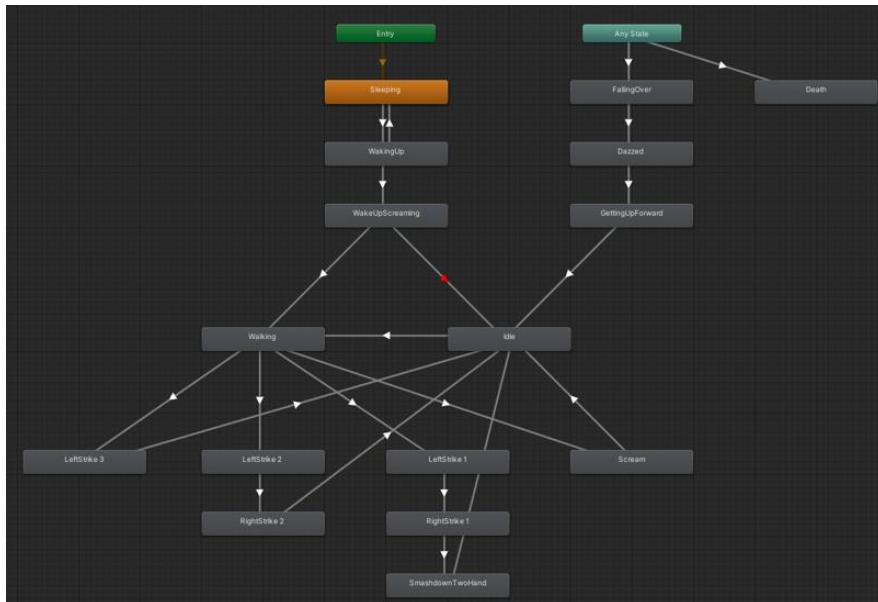
    SubHealth(damageTaken);

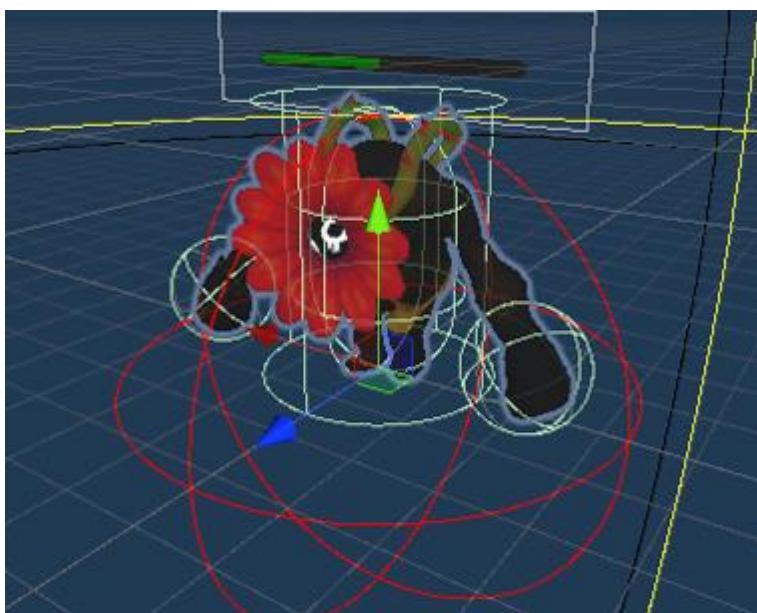
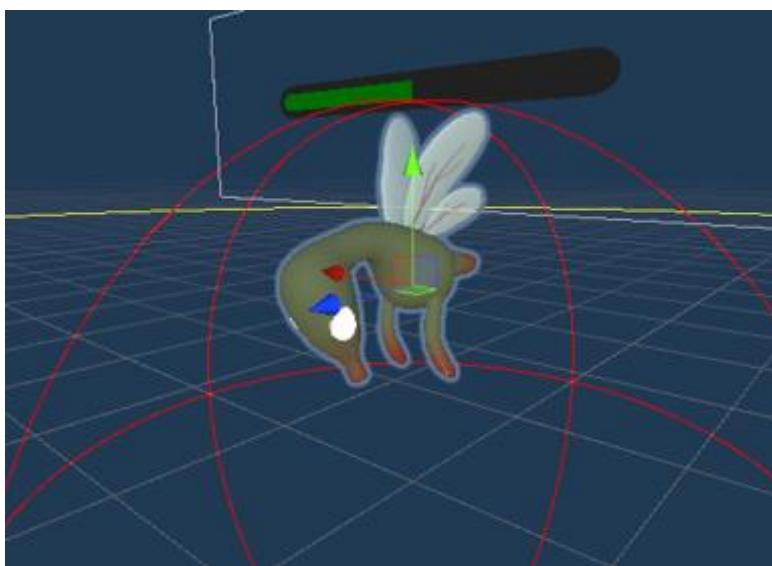
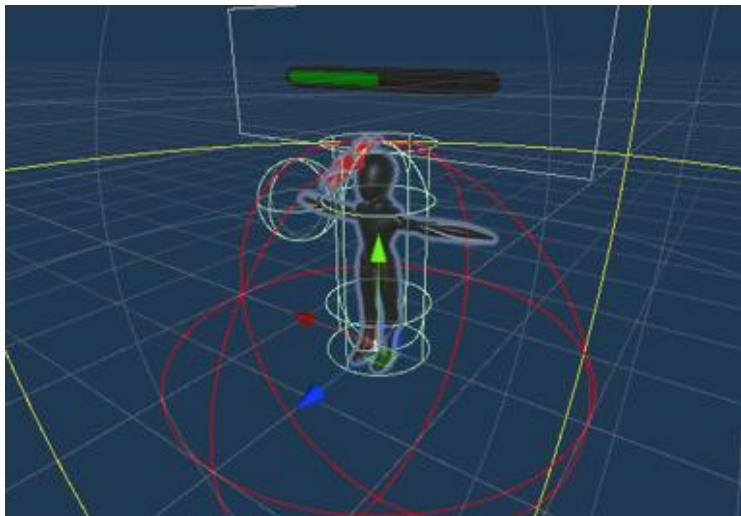
    RpcOnAttacked(damageTaken, damageType);

    onAttacked?.Invoke();
}

[Server]
[1 usage 2 JasonSkillman]
public void SubHealth(float healthAmount, NetworkConnectionToClient sender = null) {
    if(IsDead) return;
    SetHealth(health -= healthAmount, sender);
}
```

Boss states





Weapon Generation

```
 2 usages  2 exposing APIs
public class StatRange : SerializableDictionaryBase<string, Range> { };

[CreateAssetMenu(fileName = "New Weapon Part", menuName = "Scriptable Objects/Weapon Part")]
  13 usages  2 exposing APIs
public class WeaponPartTemplate : ScriptableObject
{
    public StatRange weaponStats;  ⚡ Serializable

    public List<Modifier> possibleMods;  ⚡ Serializable

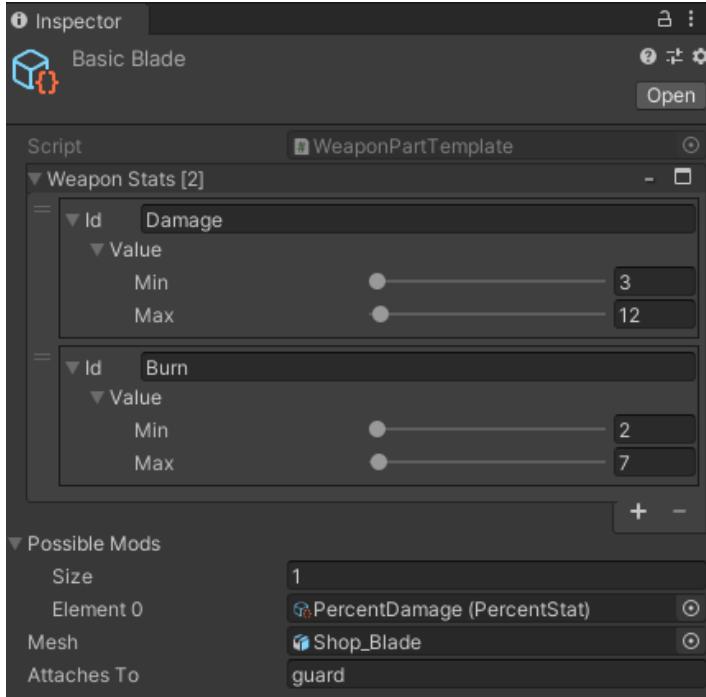
    public GameObject mesh;  ⚡ Unchanged

    public string attachesTo;  ⚡ Changed in 10 assets
}

[Serializable]
public class ModifierDetails
{
    public float weight;  ⚡ Serializable
    public Modifier modifier;  ⚡ Serializable
    public StatRange modStats;  ⚡ Serializable
}

[MessagePackObject]
[Serializable]
  2 usages  1 exposing API
public struct MountInformation
{
    [Key( 0 )]
    public string bone;  ⚡ Serializable
    [Key( 1 )]
    public Vector3 position;  ⚡ Serializable
    [Key( 2 )]
    public Vector3 rotation;  ⚡ Serializable
}

[Serializable]
  2 usages  1 exposing API
public class Range
{
    [Range(0,999)]
    public float min;  ⚡ Serializable
    [Range(0,999)]
    public float max;  ⚡ Serializable
}
```



Weapon Building

```
//Cache info
weapon = newWeapon;
//stats = weapon.stats;
meshHolder = new GameObject( name: "Mesh Holder");
var boneString = bones.ContainsKey(newWeapon.mountInformation.bone) ? newWeapon.mountInformation.bone : "default";
meshHolder.transform.parent = bones[bone];
meshHolder.transform.localPosition = newWeapon.mountInformation.position;
meshHolder.transform.localRotation = Quaternion.Euler(newWeapon.mountInformation.rotation);
var listTemplates = new List<WeaponPartTemplate>();
foreach(string partname in weapon.partNames) {
    //WeaponPartTemplate part = Resources.Load<WeaponPartTemplate>("test/*&partname");
    WeaponPartTemplate part = WeaponGenerator.LoadWeaponPart(partname);

    listTemplates.Add(part);
    GameObject weaponPartObj = Instantiate(weaponPart, meshHolder.transform);
    //WeaponPart.transform.localPosition = meshHolder.transform.localPosition;
    weaponPartObj.name = partname;
    weaponPartObj.GetComponent<MeshFilter>().mesh = part.mesh.GetComponent<MeshFilter>().sharedMesh;
    weaponPartObj.GetComponent<MeshRenderer>().material = part.mesh.GetComponent<MeshRenderer>().sharedMaterial;
}
meshHolder.transform.GetChild(0).localPosition = Vector3.zero;

for(var i = 0; i < listTemplates.Count - 1; i++) {
    try {
        var nextConnectionString = listTemplates[i].mesh.transform //Transform
            .Cast<Transform>()
            .Where(firstPart => firstPart.name == listTemplates[i + 1].mesh.transform
                .Cast<Transform>()
                .Any<Predicate<Transform>>(secondPart => secondPart.name == firstPart.name)) //IEnumerable<Transform>
                .Select(partTransform => partTransform.name) //IEnumerable<string>
                .First();

        var firstPoint : Transform = listTemplates[i].mesh.transform.Find(nextConnectionString);
        var secondPoint : Transform = listTemplates[i + 1].mesh.transform.Find(nextConnectionString);

        meshHolder.transform.GetChild(i + 1).localRotation = Quaternion.identity;
        meshHolder.transform.GetChild(i + 1).Rotate(-firstPoint.localRotation.eulerAngles);
        meshHolder.transform.GetChild(i + 1).Rotate(secondPoint.localRotation.eulerAngles);
        //GameConsole.Log(meshHolder.transform.GetChild(i + 1).position);
        meshHolder.transform.GetChild(i + 1).localPosition = (meshHolder.transform.GetChild(i).localPosition + firstPoint.localPosition) - secondPoint.localPosition;
        //GameConsole.Log(meshHolder.transform.GetChild(i + 1).position);
    } catch(ArgumentNullException ex) {
        GameConsole.Error( params args: $"Could not find connection between {listTemplates[i]} and {listTemplates[i + 1]}");
    }
}
```



Loot Generation

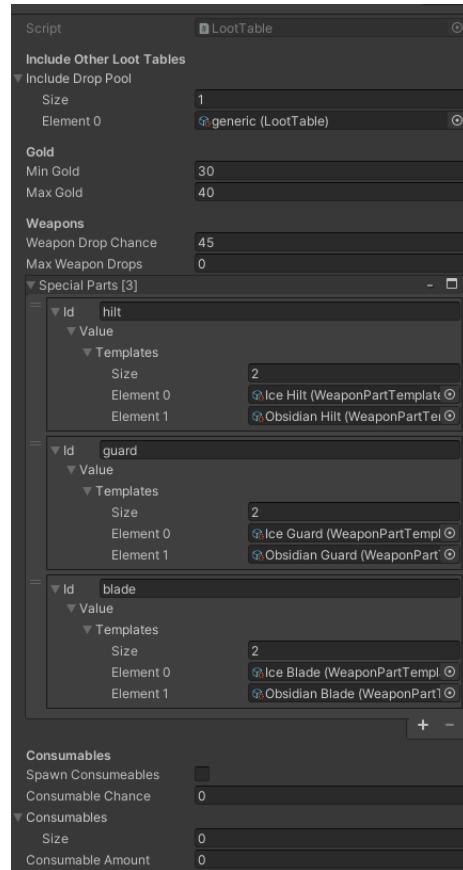
```
[CreateAssetMenu(fileName = "New Loot Table", menuName = "Scriptable Objects/Loot Table")]
[15 usages]
public class LootTable : ScriptableObject
{
    [Header("Include Other Loot Tables")]
    public List<LootTable> includeDropPool = new List<LootTable>(); // Serializable

    [Header("Gold")]
    public int minGold; // Changed in 3 assets
    public int maxGold; // Changed in 3 assets

    [Header("Weapons")]
    public float weaponDropChance; // Changed in 2 assets
    public int maxWeaponDrops; // Unchanged
    [SerializeField]
    public SpecialParts specialParts; // Serializable

    [Header("Consumables"), Tooltip("Do you want this loot table to spawn consumables?")]
    public bool SpawnConsumables; // "true"
    public int consumableChance; // "5"
    public List<GameObject> consumables = new List<GameObject>(); // Serializable
    /// <summary>
    /// How many consumables this can generate
    /// </summary>
    public int consumableAmount; // "2"
}

[Serializable]
[1 usage 1 exposing API]
public class SpecialParts : SerializableDictionaryBase<string, PartsList>
{}
```



```
    2 usages  3 exposing APIs
public class LootGenerator : NetworkBehaviour
{
    3 usages
    public static LootGenerator Instance { get; private set; }
    public WeaponGenerator wGen;  < Changed in 2 cases
    public GameObject pickUpWeapon, juansSack;  < MoneySack

    < Event function
    public void Awake()
    {
        if (Instance == null) Instance = this;
        else Destroy(gameObject);
    }

    public void GenerateLoot(List<LootTable> dropTables){...}

    1 usage
    public void GenerateLootAtLocation(List<LootTable> dropTables, Transform transform)
    {
        GenerateGold(dropTables, transform);
        GenerateConsumables(dropTables, transform);

        var itemPool = new ItemPool();
        //Generate Weapon
        var weaponDropPercentage = 0f;
        //Debug.Log(dropTables.Count);

        foreach (var table in dropTables)
        {
            //Debug.Log(table);
            weaponDropPercentage += table.weaponDropChance;
        }

        var wDropChance int = Random.Range(0, 100);
        if (wDropChance < weaponDropPercentage)
        {
            itemPool.FeedTemplates(dropTables);
            spawnWeaponAtLocation(itemPool, transform);
        }
    }
}
```

```
 1 usage
void SpawnWeaponAtLocation(ItemPool itemPool, Transform transform)
{
    var gWeapon :GameObject = Instantiate(pickUpWeapon, transform.position, transform.rotation);
    var weapon = wGen.GenerateSword(itemPool);
    gWeapon.GetComponent<PickupWeapon>().weapon = weapon;
    var rb = gWeapon.GetComponent<Rigidbody>();
    //TODO: use variable, don't hardcode
    rb.AddForce(new Vector3( x: Random.Range(100, 150), y: Random.Range(100, 150), z: Random.Range(100, 150)));
    NetworkServer.Spawn(gWeapon);
}

[Server]
 1 usage
public void DropWeaponAtLocation(Weapon weapon, Transform transform)
{
    var gWeapon :GameObject = Instantiate(pickUpWeapon, transform.position, transform.rotation);
    gWeapon.GetComponent<PickupWeapon>().weapon = weapon;
    var rb = gWeapon.GetComponent<Rigidbody>();
    //TODO: use variable, don't hardcode
    rb.AddForce(new Vector3( x: Random.Range(100, 150), y: Random.Range(100, 150), z: Random.Range(100, 150)));
    NetworkServer.Spawn(gWeapon);
}

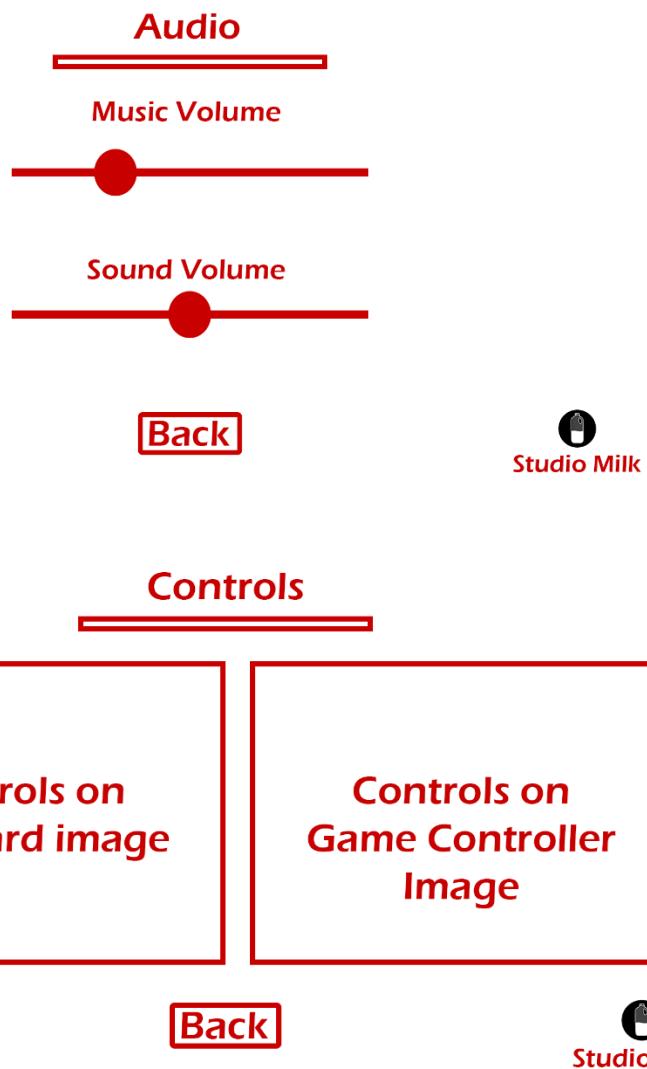
[Server]
 1 usage
public void GenerateGold(List<LootTable> dropTables, Transform transform)
{
    int totalMinGoldAmount = 0;
    int totalMaxGoldAmount = 0;

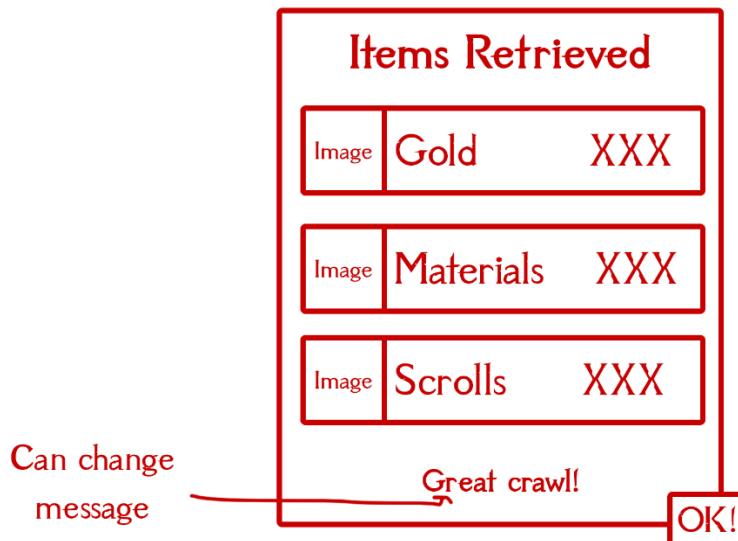
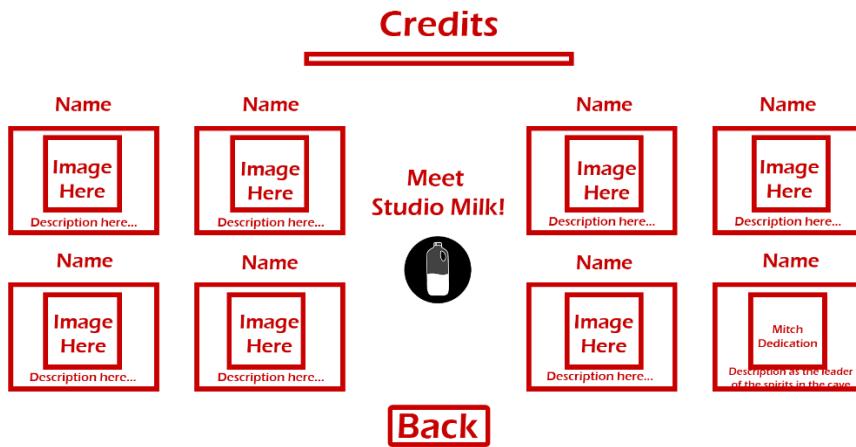
    foreach (var table in dropTables)
    {
        totalMinGoldAmount += table.minGold;
        totalMaxGoldAmount += table.maxGold;
    }

    var sack :GameObject = Instantiate(juansSack,
        position: new Vector3(transform.position.x, y: transform.position.y + 1, transform.position.z), transform.rotation);
    sack.GetComponent<MoneySack>().SetMoney(UnityEngine.Random.Range(totalMinGoldAmount, totalMaxGoldAmount));
    sack.GetComponent<Rigidbody>() // Rigidbody
        .AddForce(new Vector3( x: Random.Range(100, 150), y: Random.Range(200, 250), z: Random.Range(100, 150)));
    NetworkServer.Spawn(sack);
}
```

UI

Wireframes

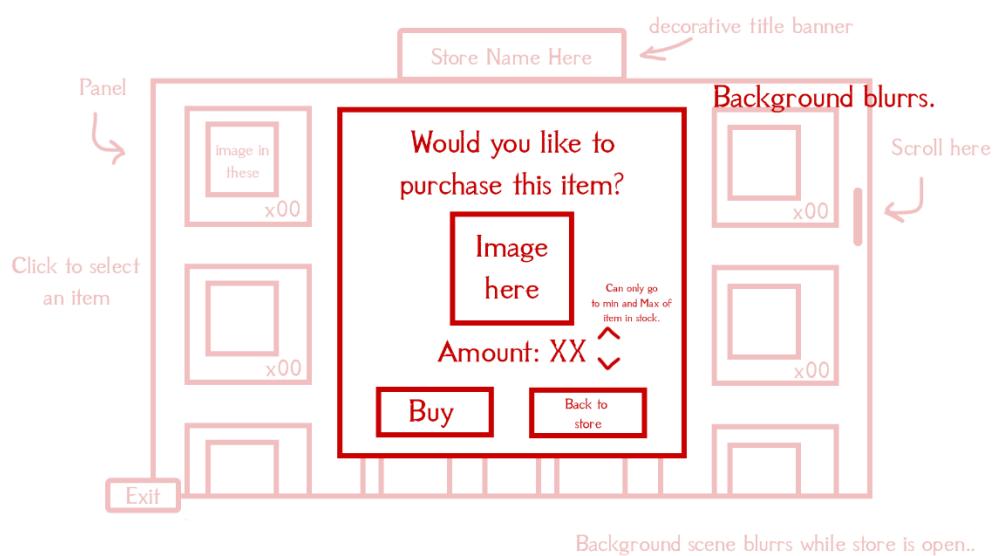
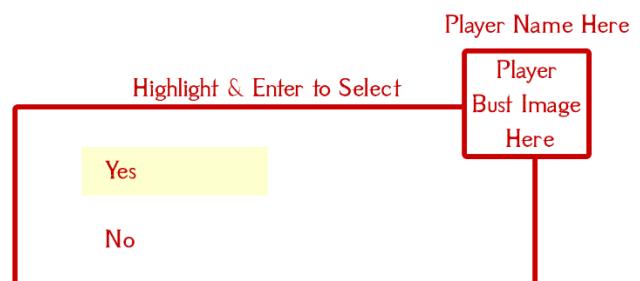


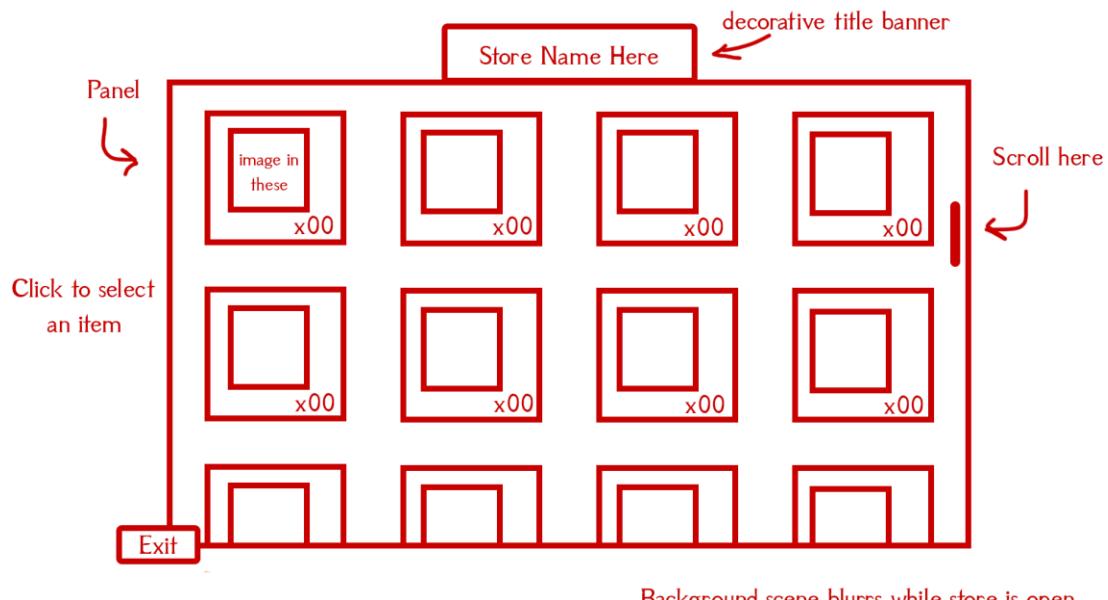


Note: The background to the loading screen will be different concept art pieces by Dorian. These loading images will be random out of a selection of different art pieces.

Loading...

Tip #XX: Write an awesome tip here to help the player read something while passing the time of waiting here... blah blah blah

**Back**



Background scene blurs while store is open..



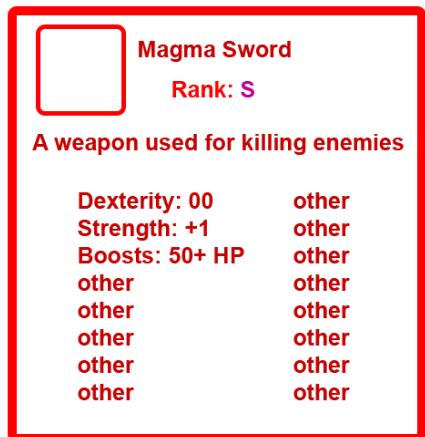
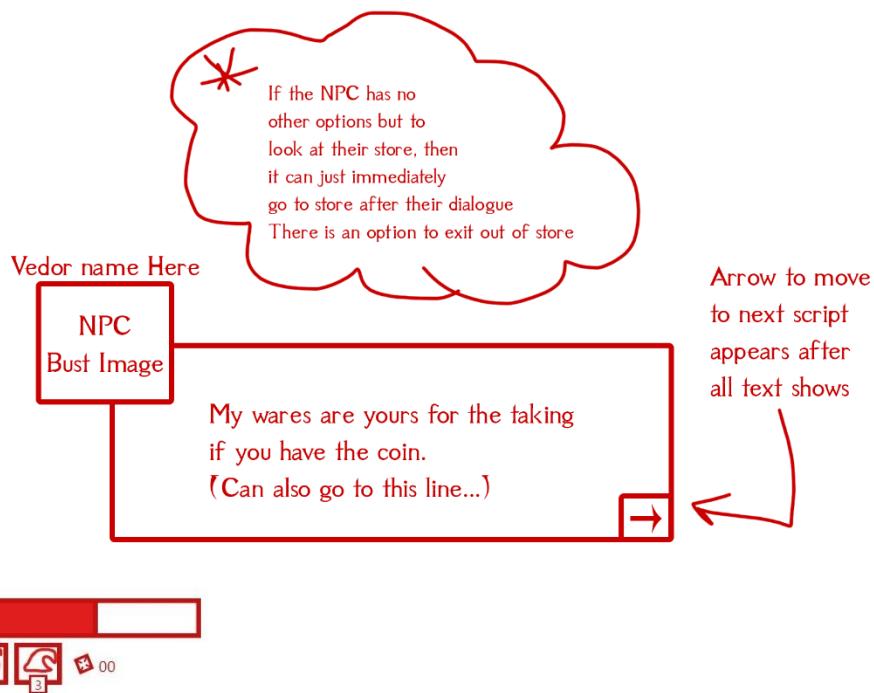
Start

Quit

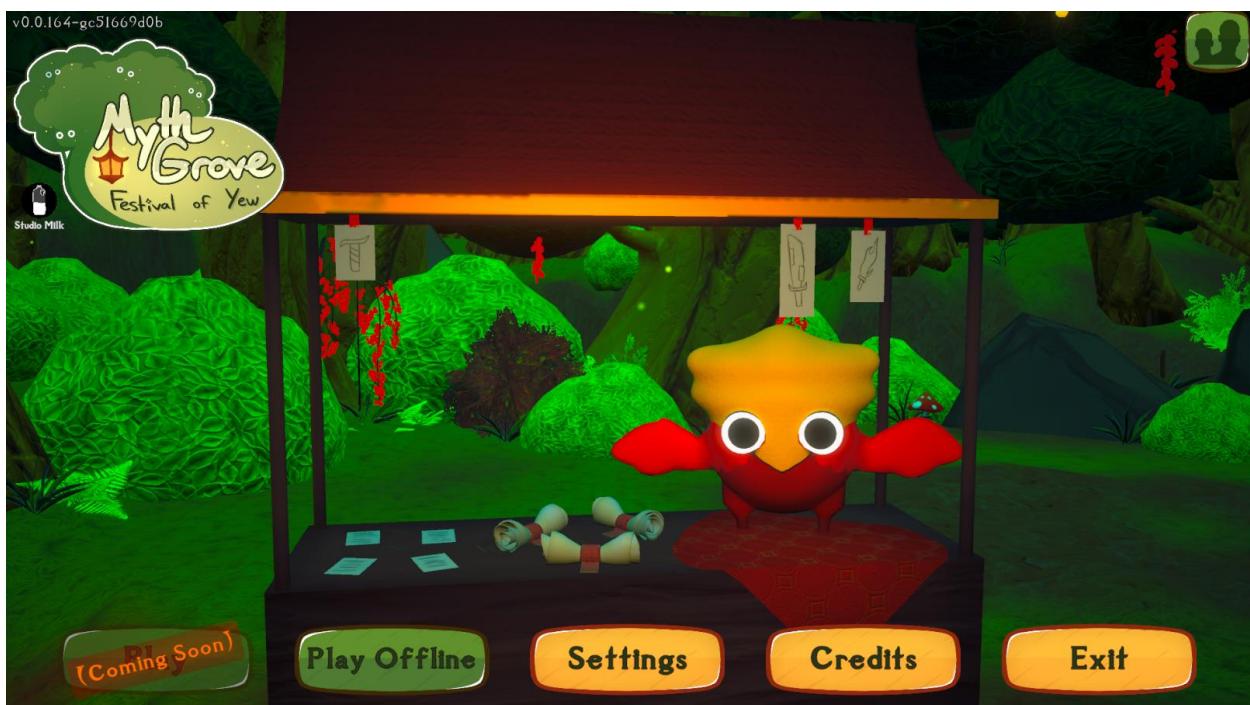
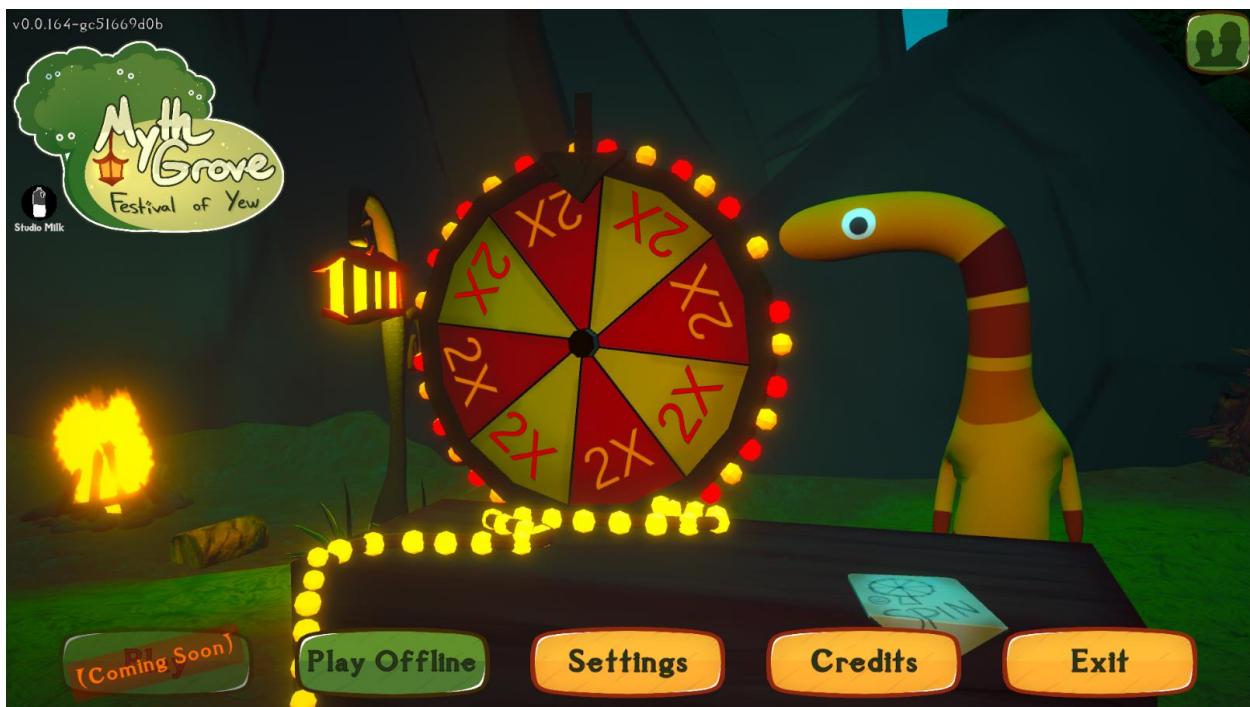
Version # Here

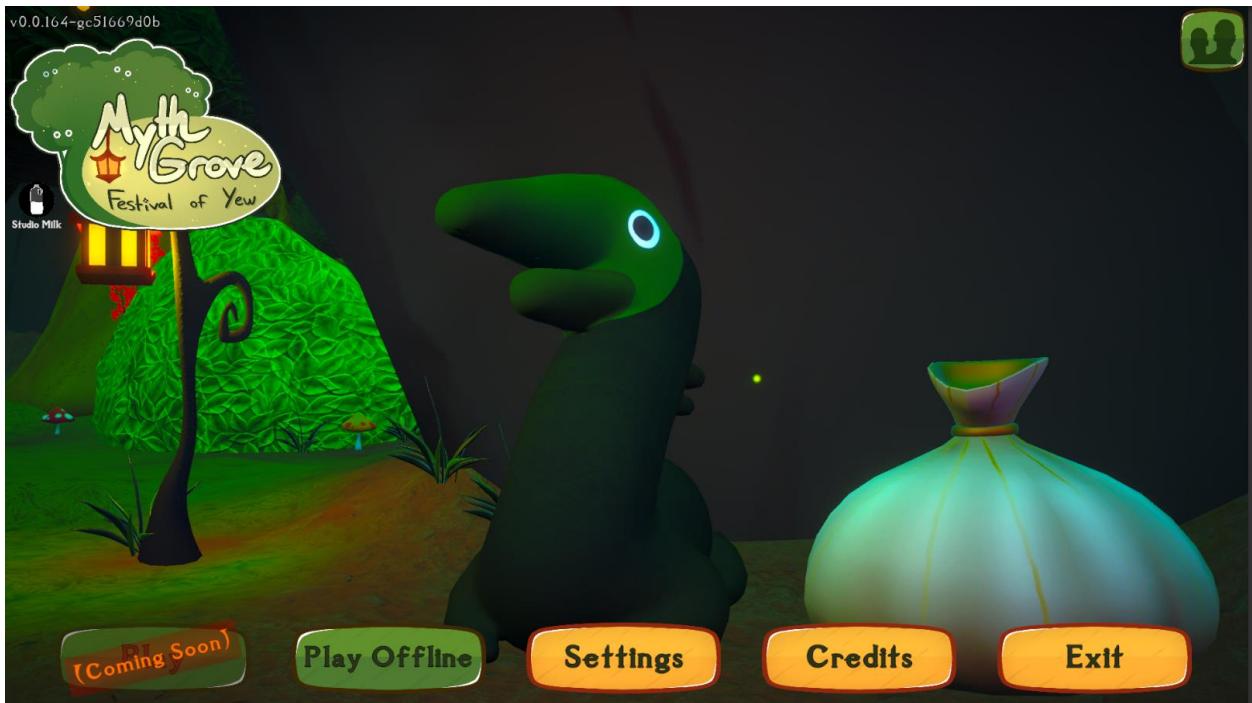
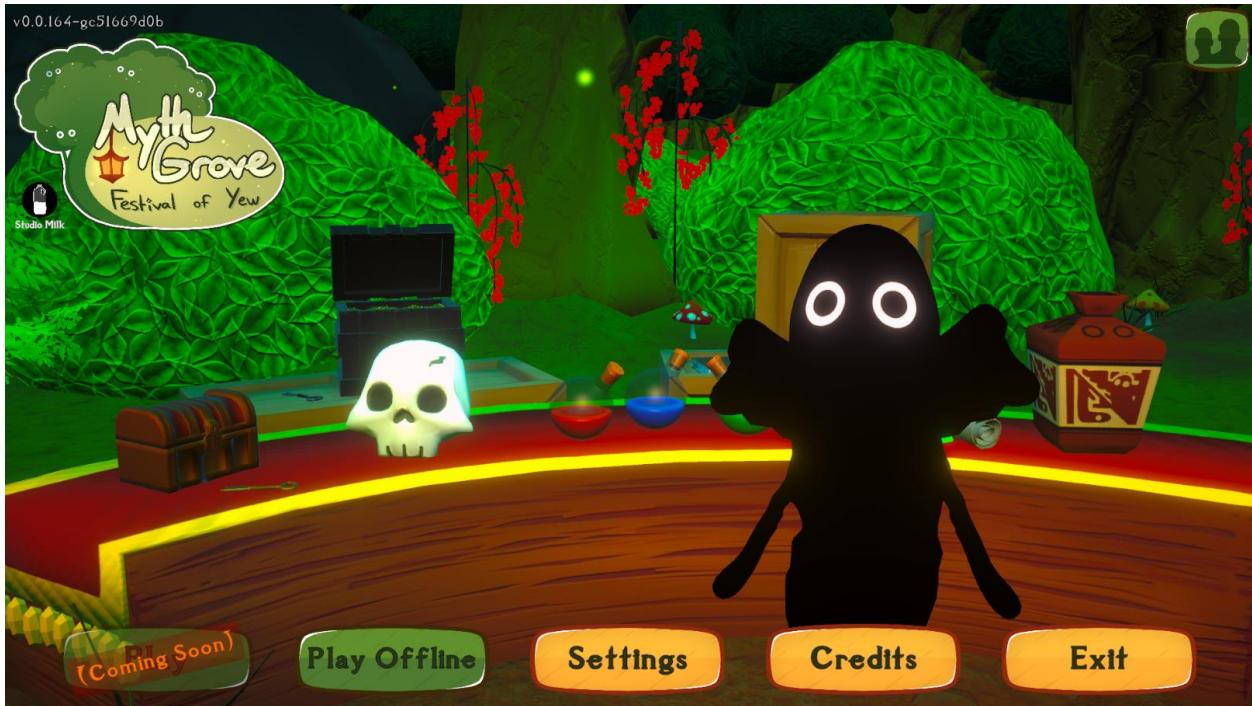
Settings/options





Main Menu





Credits

Credits

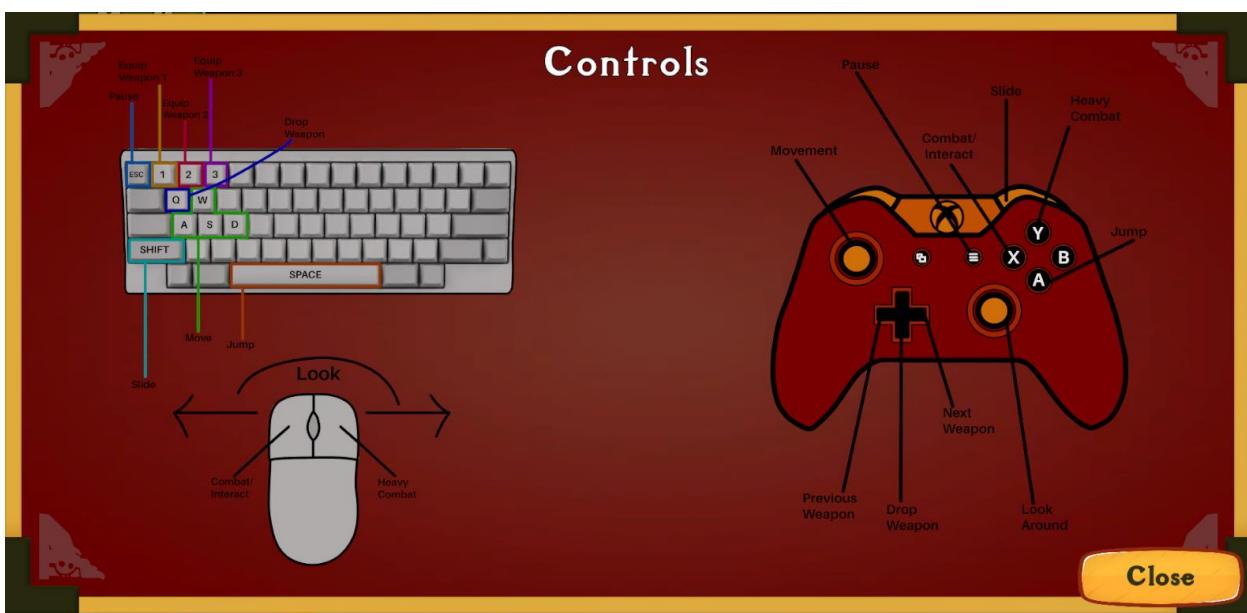
Jason Skillman  Project Manager, Room Builder, Level Design, Networking, AI & State Machine, & Combat System	Amanda Coler  Co-Project Manager, Lead UI/UX, Wireframing, Reward Schedule, 3D Plant Models, Level Design, Sound Designer, & Game Documentation	Joel Hanson  Lead Programmer, Inventory, Interactables, Weapon Generation, Early Statemachine, & Weapon Switching	Juan Santos  3D Sculpting, 3D Modeling, Rigging and Animations, 3D Environmental Models, & Texturing
Kyle Gray  Lead Networking, Player Controller, Player Attack, Weapon Modifiers, UI System,	Bell Wiekman  Lead Designer, Lead Sound Designer, Lead Character Artist, 3D Environmental Texturing, Color Palettes, & Music Composition	Christian Martin  Narrative Story Writer, Weapon Modeling & Texturing, Level Design, 3D Environmental Models & Texturing.	Jeff Knapp  Music Composition

Meet

Studio Milk

Back

Settings



Player Inventory & Equipment



Interactions





Dialogue

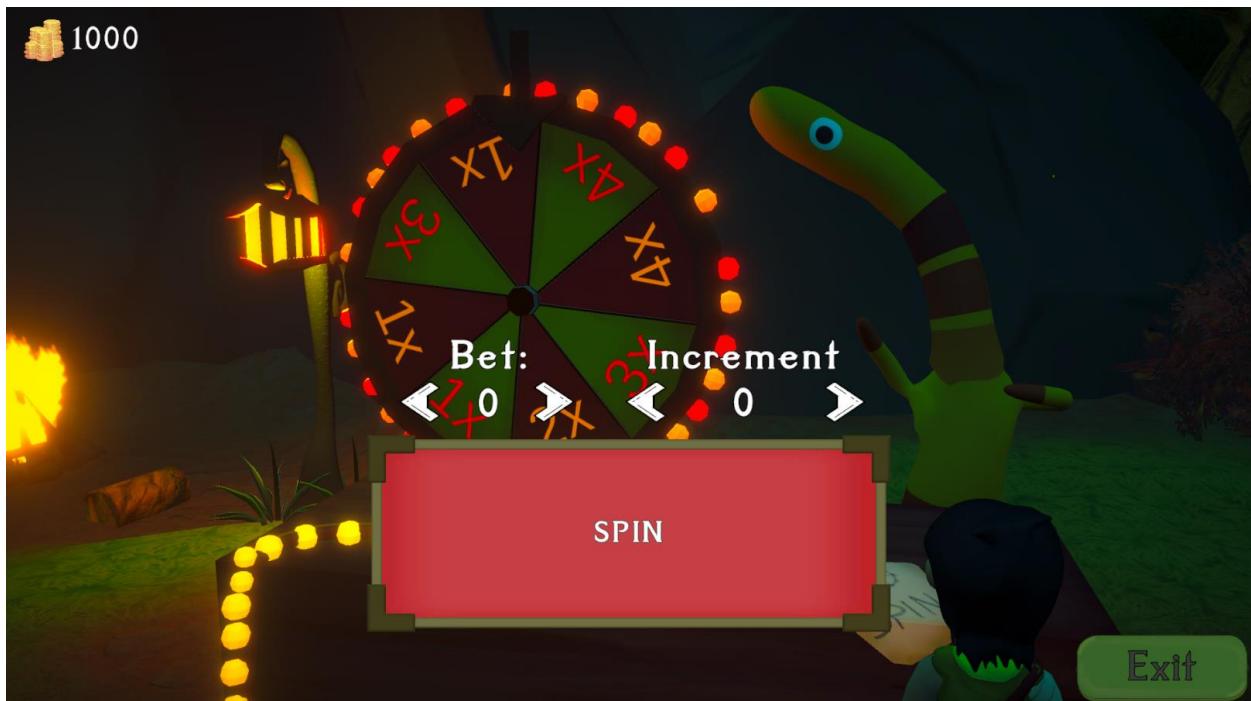






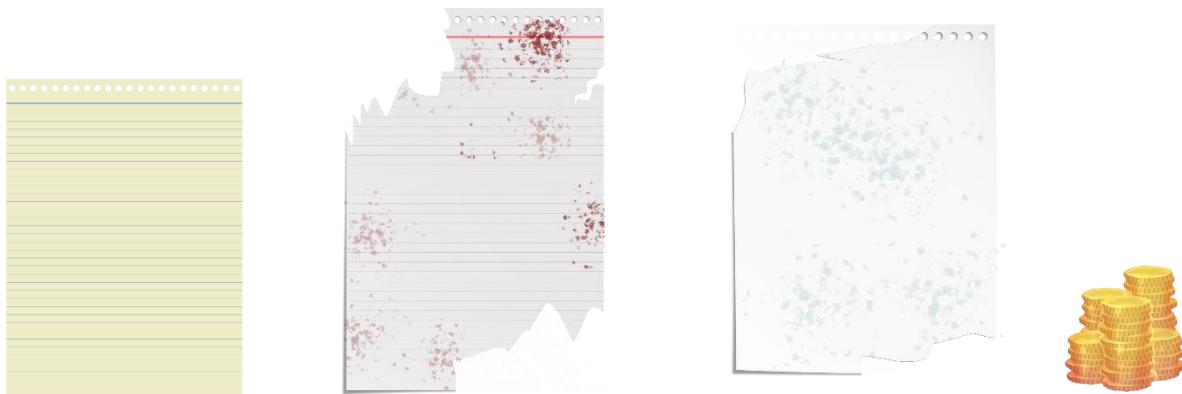
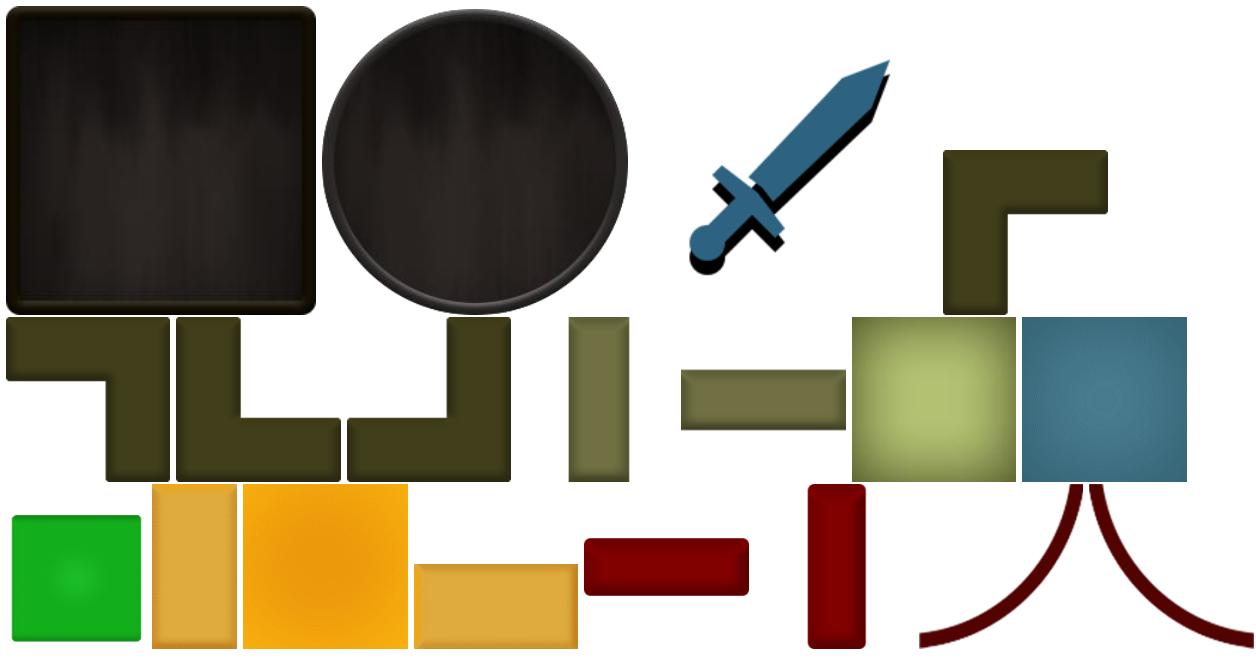
Store UI

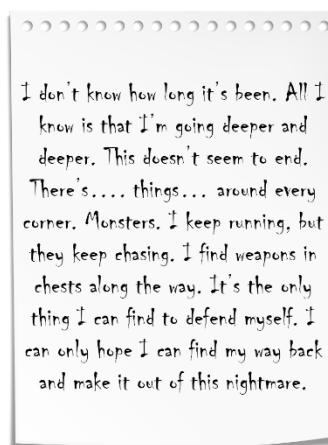
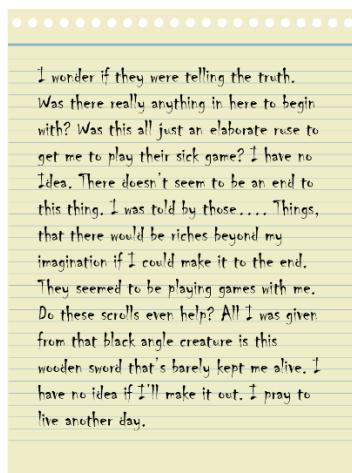
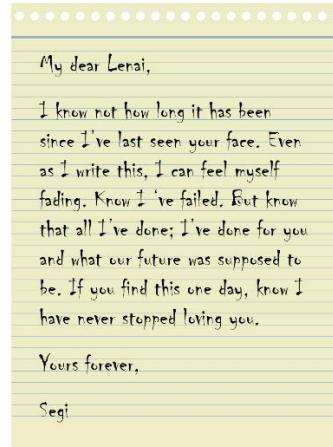
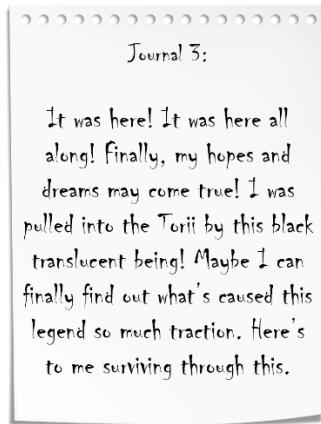
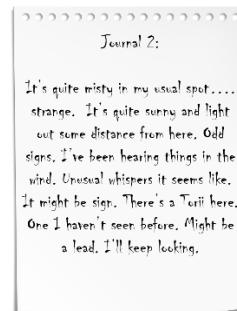
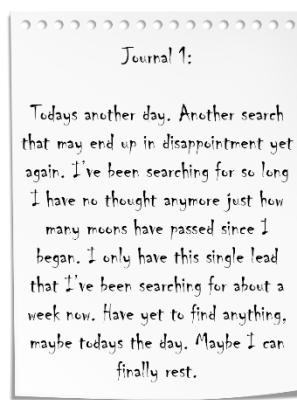
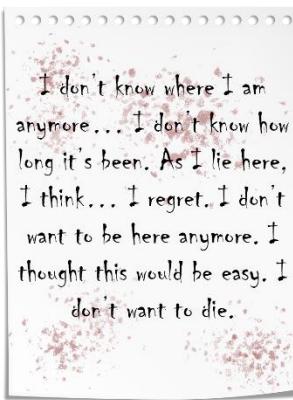


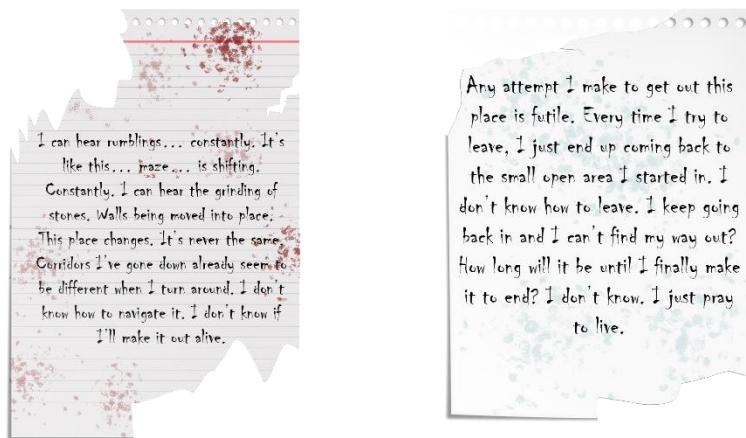


Sprites & Assets









Assets

Font

Aetherius

Example:



but those that enter never return.

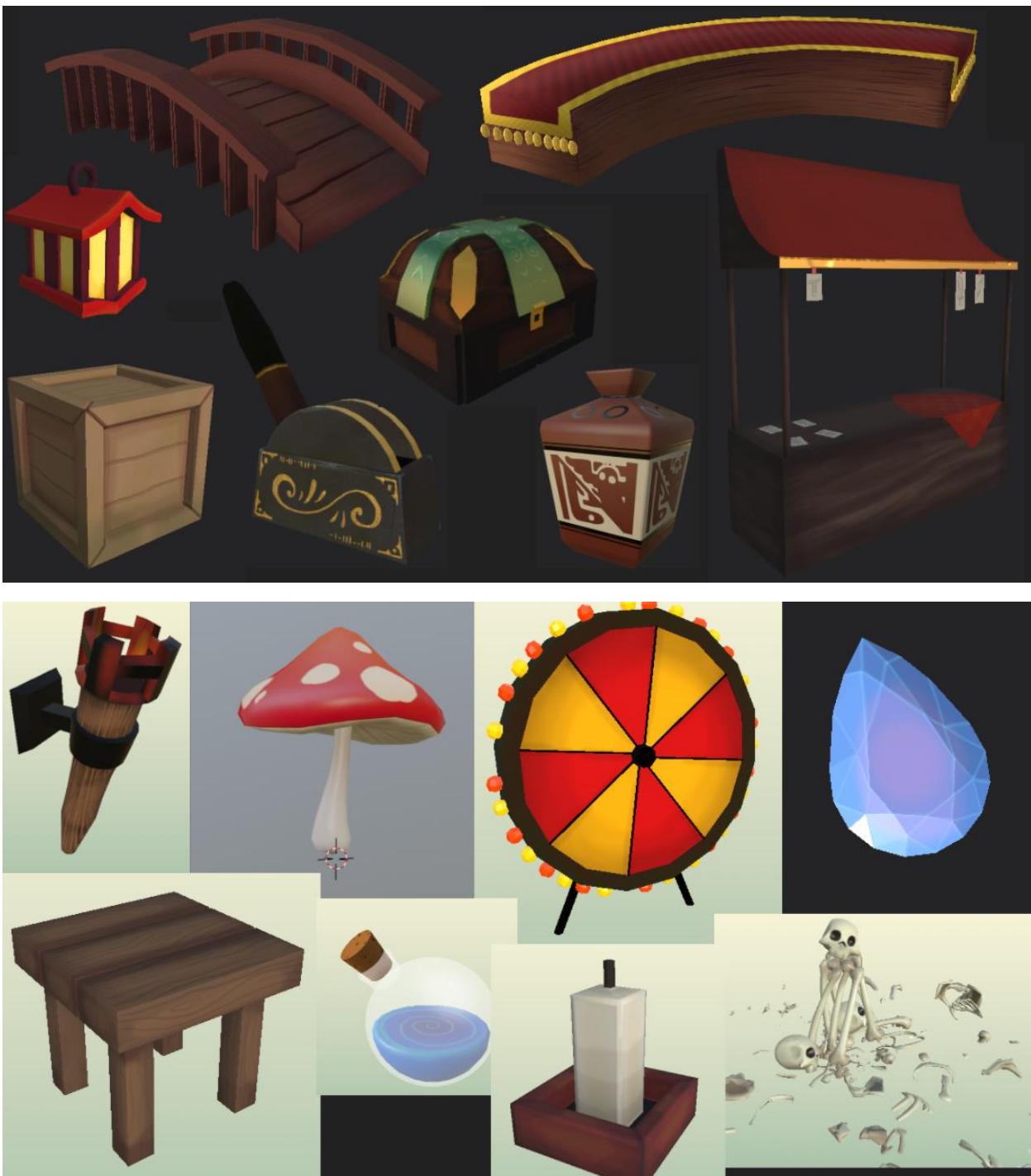
Sound & Effects

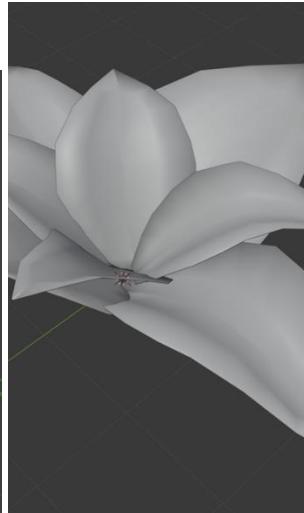
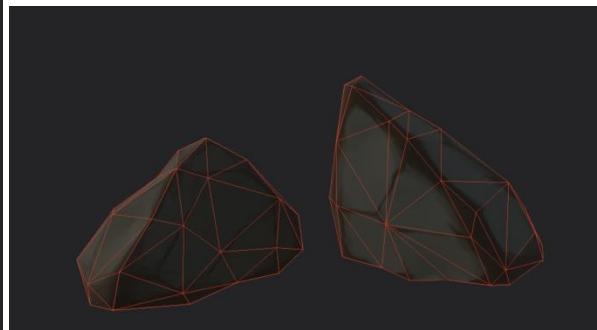
- 🔊 AmbientHUBSound...
- 🔊 AmbientMossyDun...
- 🔊 BarrelBreak.mp3
- 🔊 BoneDropCrush.mp3
- 🔊 ChestOpen.mp3
- 🔊 FireFlickerSound.mp3
- 🔊 GateOpenSound.mp3
- 🔊 GoldPickup.mp3
- 🔊 GoldPickupSound2....
- 🔊 HealthUp.mp3
- 🔊 ItemPickupSound....
- 🔊 MonsterDeathSoun...
- 🔊 MonsterDeathSoun...
- 🔊 MonsterIdleSound_...
- 🔊 MonsterIdleSound_...
- 🔊 PlayerDeathSound....
- 🔊 PlayerHit.mp3
- 🔊 PortalSound.mp3
- 🔊 PotionDrink.mp3
- 🔊 PotionPickup.mp3
- 🔊 PullLeverSound.mp3
- 🔊 ScrollOpen.mp3
- 🔊 SlimeDeathSound....
- 🔊 StepSound.mp3
- 🔊 SwordSing2.mp3
- 🔊 SwordsMetalClash....
- 🔊 SwordSwing.mp3
- 🔊 TeleportSound.mp3
- 🔊 VineWhipSound.mp3
- 🔊 WeaponEquipNew....

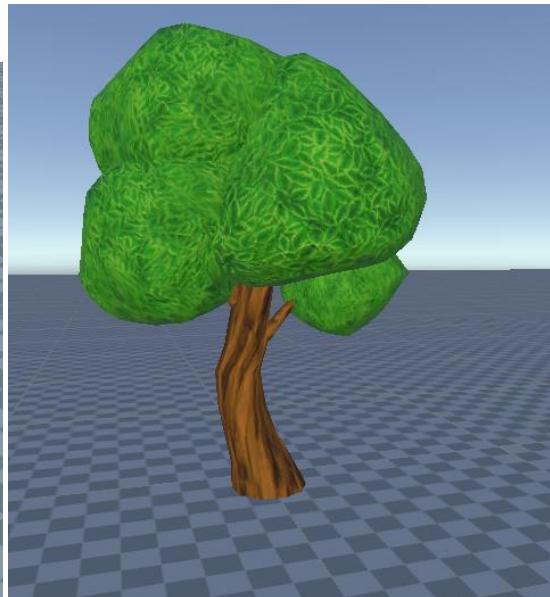
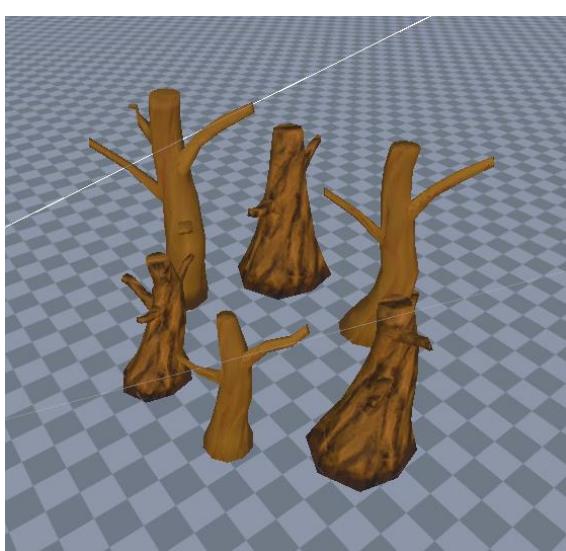
Particle Effects

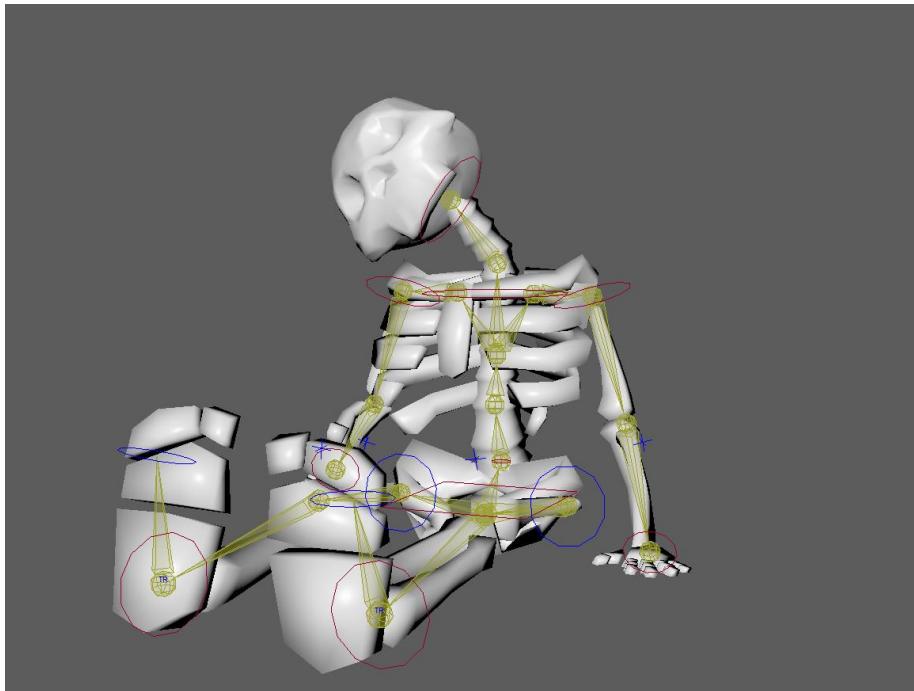


Environmental Art

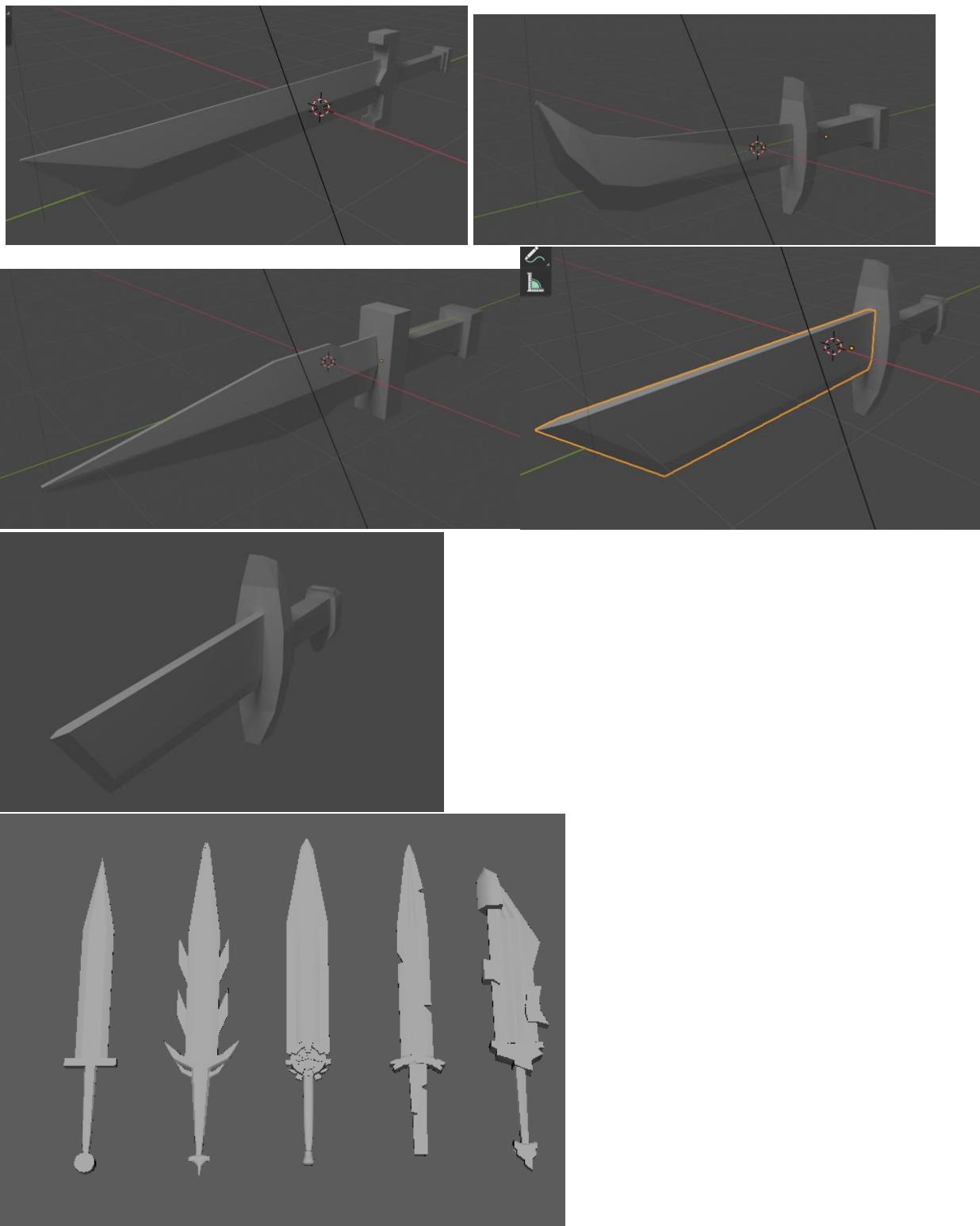








Swords





Character Models

Player

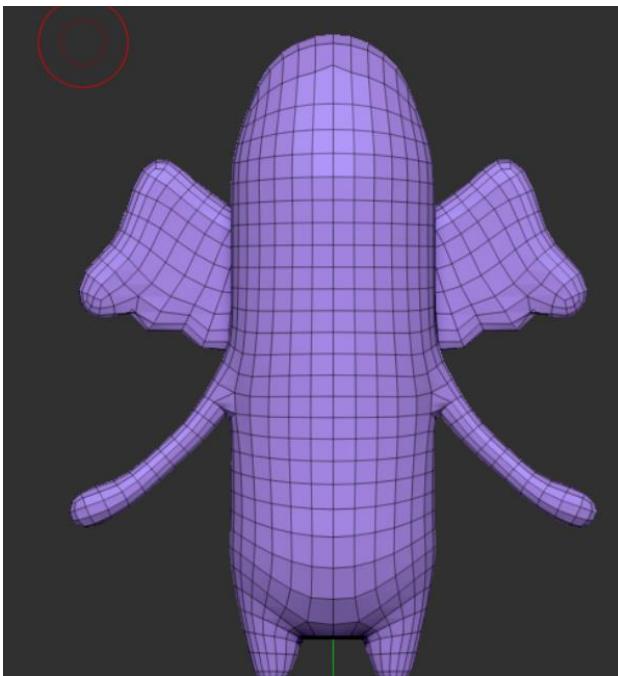




NPC Models

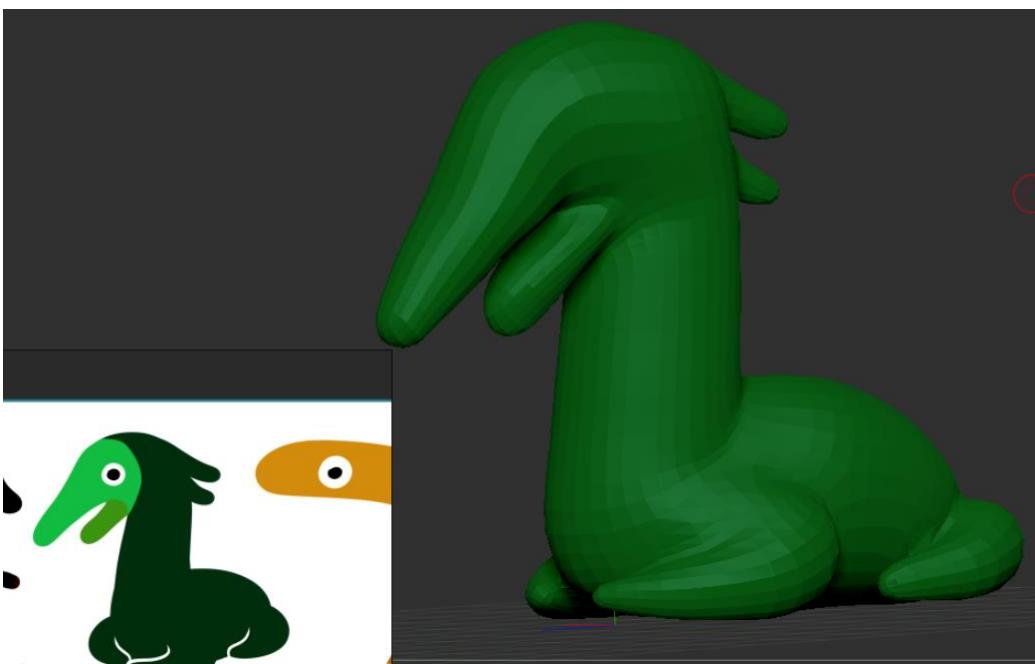


Mamo

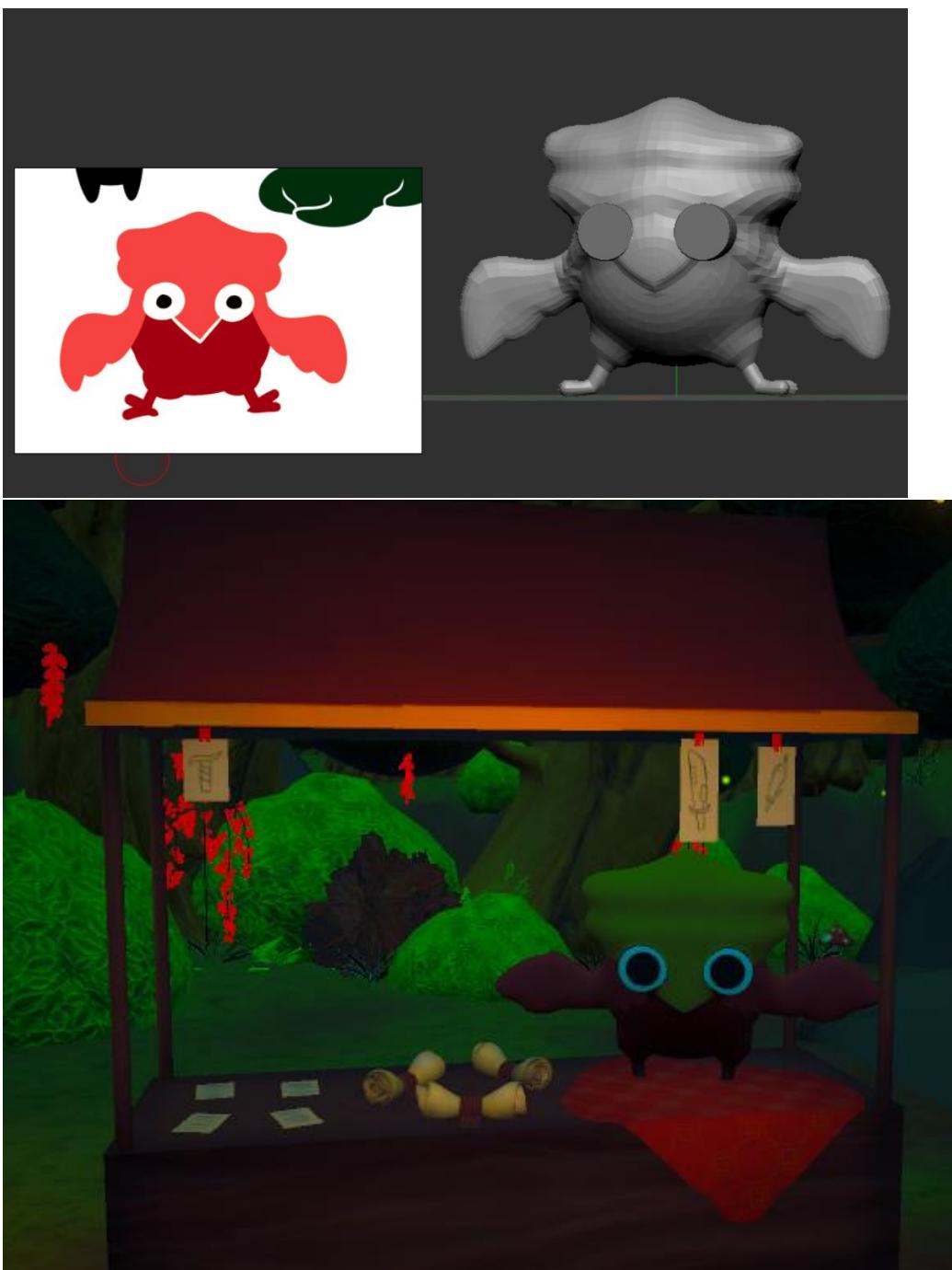




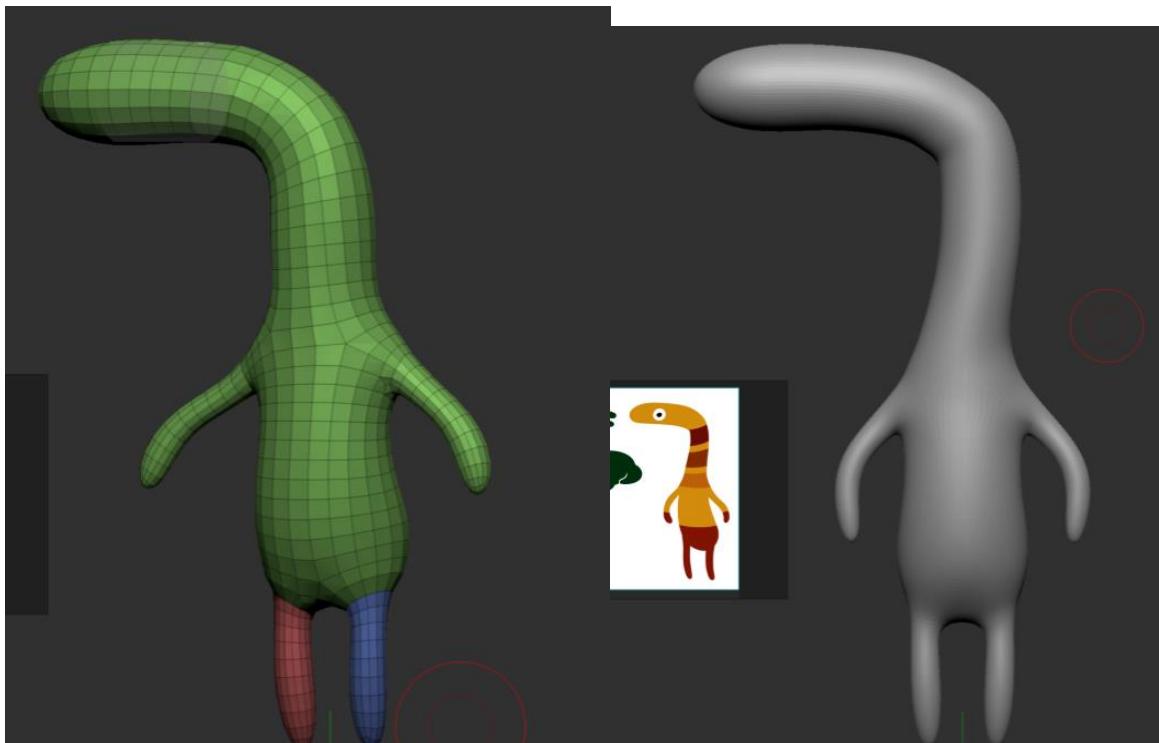
Seawee



PikPok



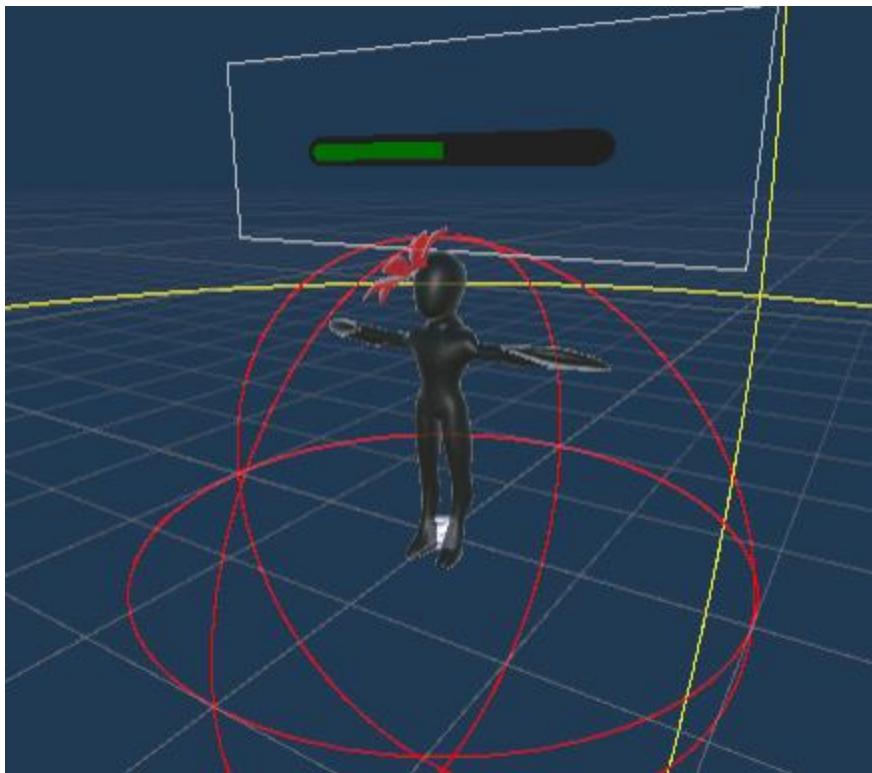
Randy



Monsters



Vinemancer





Bugman

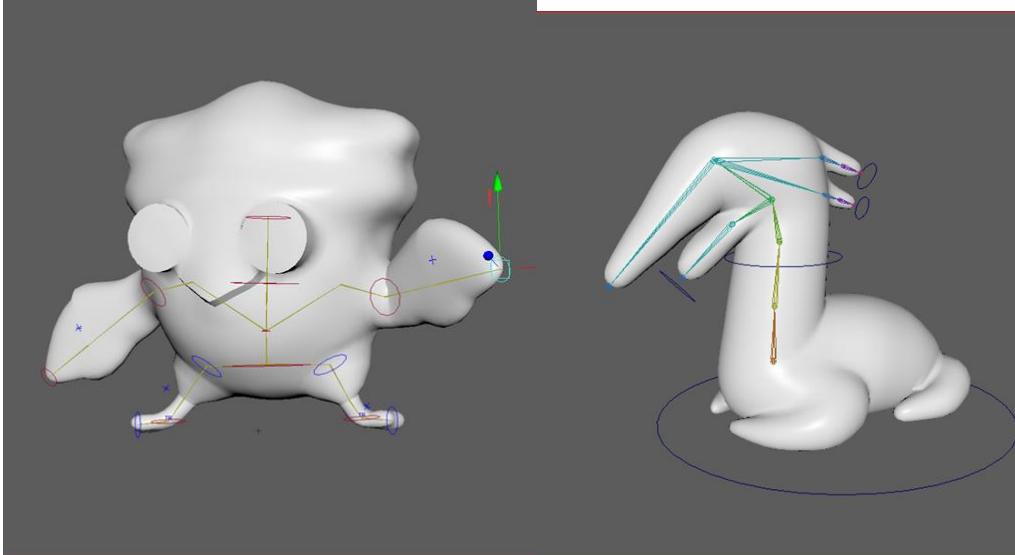
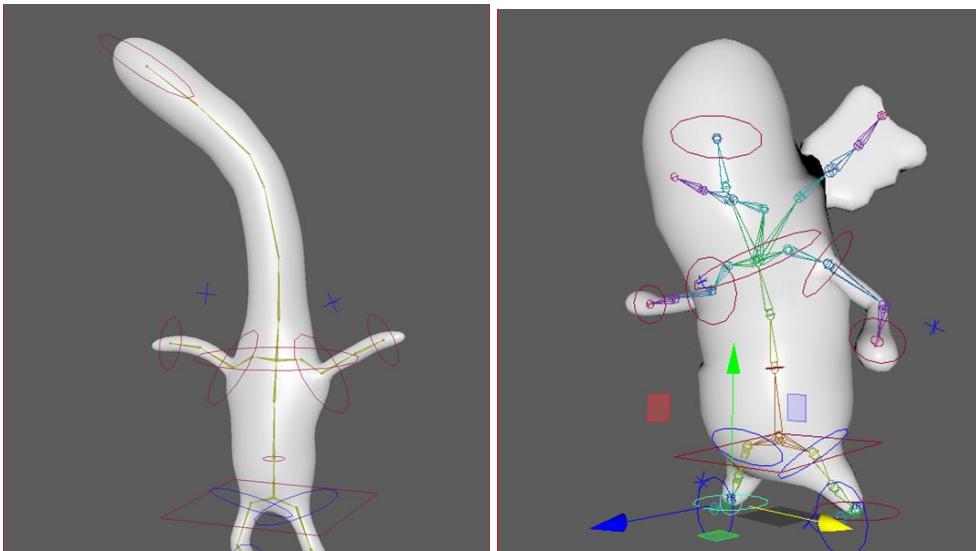


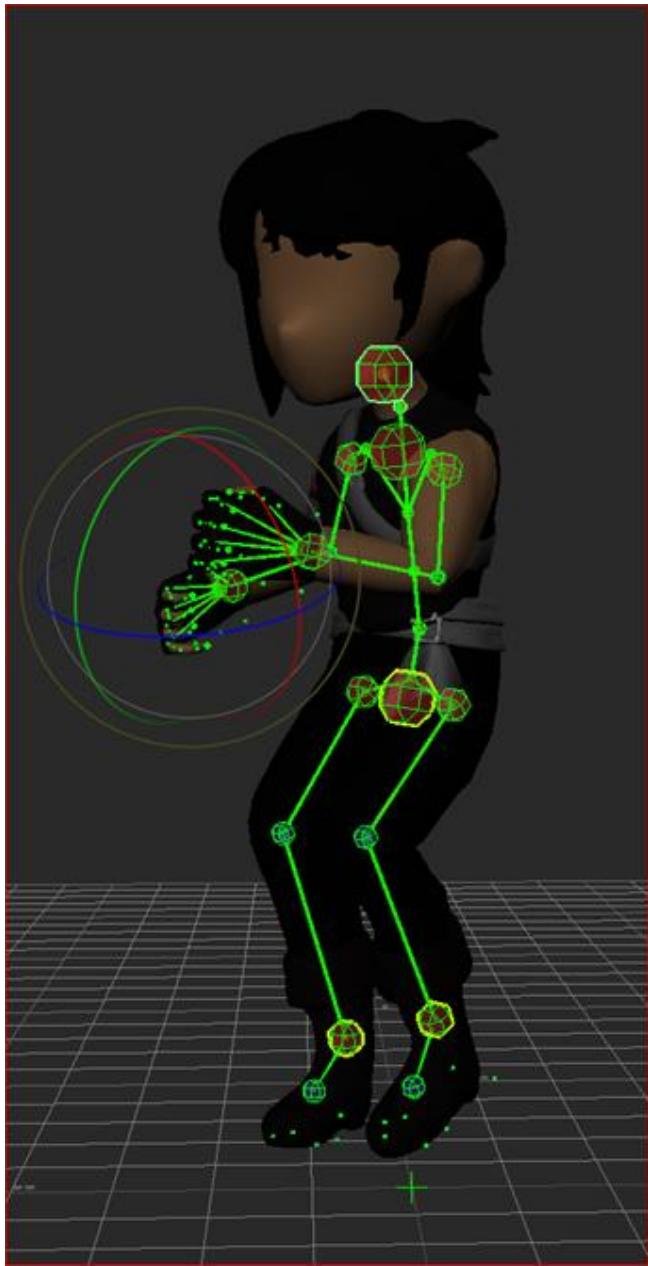
Boss

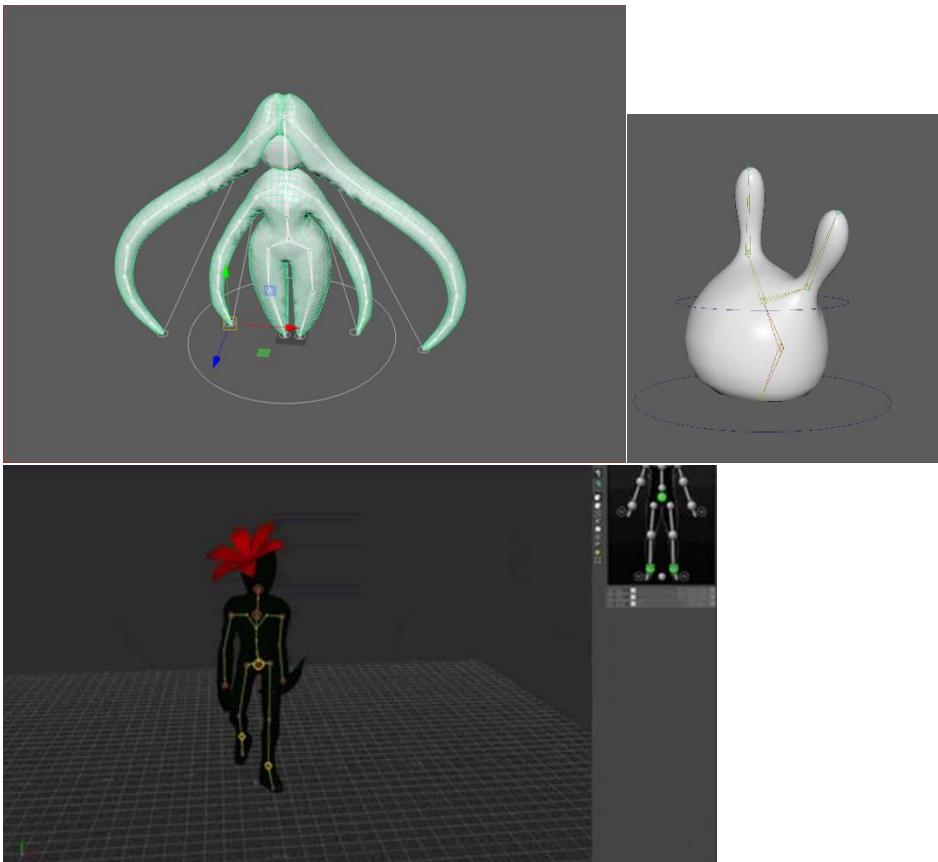


Animations & Rigging

NPC rigs







Narrative Design

Backstory

The legend had been passed down for years. Tales of hero's and seekers of fame looking to get their hands on a treasure so vast, no one knows how much is there. I myself am one of these "Seekers." I've been on the trail of this fortune for many moons now, but I fear that it might be just that, a legend. This day I happened to stumble upon something strange. A Torii where one normally isn't in my usual hikes around the place where the legend is supposedly whispered. I walk to the Torii and put my hand to it. Suddenly, a bright yellow light flashed before me, and as I look to the Torii I notice, its inner space is covered in the yellow light. I walk up to the light, and just before I put my hand to light. I am grabbed by a black, translucent stub of a hand. A small round head with large, glowing eyes pops through the portal.

"Oh, another one! Welcome! Come, come there's much for you to know."

I get pulled through to somewhere I've never been. The aura here feels... unnatural. Though it looks like many other forests around the area, there is an air of spirits about. I know now that the tales are true. Many have tried, many have failed, and almost no one has come back alive. This time, it's my turn, and I won't be turning back.

Dialogue

- "Browse until your hearts content"
- "My wares are yours for the taking if you have the coin"
- "What wonders can I bring you today?"
- "Have you seen my shop lately? Just restocked!"
- "If you don't have the cash for it, leave, and come back when you do."
- "No coin? Come back when you do, my items will be available!"
- "Just what you were lookin for, Huh?"
- "Ohhh you're gonna enjoy that one!"
- "Stocked and ready to rock your world!"
- "Step right up, Step right up! The wheel doesn't hold a grudge!"
- "Spin! Spin! Spin! Where's she gonna go? Only luck will know!"
- "My, My! What a lucky person I've happened upon today!"
- "Oh no! Bad luck old chum! Better luck next time!"
- "Hmmm, some coin not too shabby for the other vendors around."
- "Wanna give it a shot? You can only win if you know how to spin!"
- "Try again? The wheel may just be on your side!"
- (Player to NPC) "Lets Spin it Right Round Baby!"
- (Player to NPC) "I'm not feeling so lucky right now."
- (Player to NPC) "Let's give it another go!"
- (Player to NPC) "Maybe another time."
- "Wanna see what I got? Or maybe you'd like to align your stars?"

- “Skills and thrills! That’s a good one!”
- “The mystery of the scroll never ends”

Mamo NPC (C-Shaped Table)

- “Have you seen my shop lately? Just restocked!”
- “Just what you were lookin for, Huh?”
- “Stocked and ready to rock your world!”
- “Ohhh you’re gonna enjoy that one!”
- “What wonders can I bring you today?”
- “Name’s Mamo! What can I do for ya?”
- “Look what I found! Something that might interest you!”
- “Something creepy in this one, might wanna take a look.”

Mamo Story Side Dialogue

- “Well I was hoping you wouldn’t ask this.”

Randy NPC (Wheel Booth)

- “Whatever.”
- “If you don’t have the cash for it, go get some more.... I’ll be here, as always.”
- “Hi, name’s Randy.”
- “You again? Sheesh.”
- “Come one Come all! Spin the wheel of fortune! ‘Not that it does you any good.’”
- “Leaving? Do whatever you want.”
- “You’ll be back... Maybe.”

Seawee

- “...”
- “... wee”
- “See?”
- “Seawee Sea.”

Pikpok

- “Take a look! Please? Hopefully it’s what you’re looking for.”
- “Spiffing!”
- “Maybe C-C-Come back? I’ll be here!”
- “This should help! I think, I don’t know, maybe?”
- “Thank you! Oh this is marvelous!”
- “Scrolls can be of great use!”
- *Insert Astronomy fact here:*
 - On Mercury a day lasts twice as long as a year
 - One teaspoonful of neutron star would weigh the same as the entire human population
 - There are stars we will never be able to see

- If Jupiter's magnetic field were visible, it would appear bigger than the Moon in the night sky
- Neptune has only completed one orbit around the Sun since its discovery
- Hot ice is a thing
- One million Earths can fit inside the sun.
- You can't walk on Jupiter, Saturn, Uranus or Neptune because they have no solid surface.
- Olympus moon, which is 3 times higher than the Mount Everest, is the highest mountain known to man and is located on Mars.
- The sunset on Mars appears blue.
- Driving a car to the nearest star at 70 mph would take more than 356 billion years.
- It would take nine years to walk to the moon.

Chosen Narratives

- Some of the narratives were designed with intent to create doubt or discomfort for the player. The player would pick up documents that had been written as "last testaments" for the other persons who had entered the dungeon before the player.
- Others were chosen by lines/descriptors given to me in order to create other scenarios or feelings for the player

What They Are

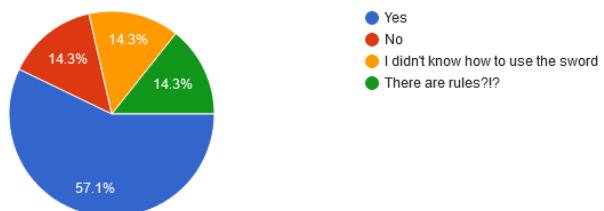
- The narratives themselves encompass the feelings or thoughts of characters made up to be persons outside or entering the dungeon before the main character in the game. They each have different personalities and themes in order to show the player that many people from many different timelines and places around the world could get to the place that he has in now.
- These narratives also serve to create the short, openness of story elements to help the player in connecting more to the world and want to continue playing the game and learn more.

Conclusion

Player Feedback

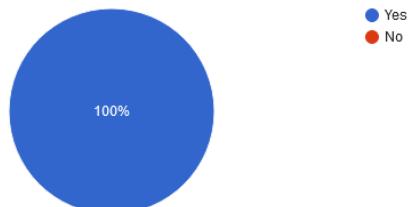
Are the rules easy to understand?

7 responses



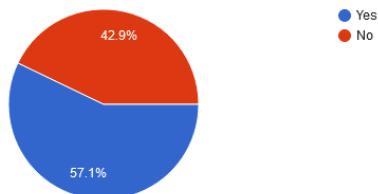
Was the length of game-play appropriate?

7 responses



Could you see yourself playing this game again?

7 responses



What did you like most about the game?

7 responses

Art style

The art style and presentation.

I loved the art style and aesthetic

How much has been done in 10 weeks

Nice looking environments, cute character designs

moon walking, character designs

The music and the characters are all so sweet

What did you dislike about the game?

7 responses

The combat was too simple, as it was really just trading damage

A lot of key mechanics don't work. The camera was also a bit too sensitive.

I don't think any of the mechanics worked, and there was no clear way to progress.

Turning was rather quick and abrupt

Combat system feels like it needs a lot of work, overall game felt a bit generic

Attack speed is too slow, UI did not show up for inventory, weapon slots, or randy's dialogue prompt, quick tips were too quick

The dungeon itself was a bit boring to go through but I think it was only cause I couldnt pick up stuff

Which game play mechanic stood out to you the most?

7 responses

It's very put together and nice to look at.

Not a whole lot.

My favorite part is the part at the beginning with the interesting creatures.

How much has been done in 10 weeks

Getting different loot, specifically getting cool new swords. Cute NPCs

gambling

I think I played with the wheel for about 20 minutes before I went into the dungeon

How would you describe this game to your friends?

7 responses

Bro, it's like this dungeon game and you can play together and you loots stuff wow epic

Has the makings of a good game.

The game is charming but needs a little work, definitely something to keep an eye on though!

Kill feminine black cleaver handed people, get loot, talk to merchants with scoliosis. Or that it's a dungeon crawler with online capability

Dungeon Crawling Rouge-like

hack and slash

Rouge lite dungeon crawler similar to wizard of legend

What did you find most confusing about the game?

7 responses

I'm not entirely sure what the money is useful for aside from buying loot at the shop, when that's more implemented.

The wheel, I guess?

I was not sure how to progress. There is no clear indication on what is going on and what you are meant to do, so you end up wondering aimlessly a lot. I know the point is to get lost, but I often found myself just running a bunch and not paying attention to anything else.

The controls. At least without going through menus

Where do I go in the dungeon? Might need a map implemented to remedy this

gambling and my swords didn't work

Nothing confused me a noticeable amount

Did you encounter any bugs? Write them here.

7 responses

with 4000, but suddenly was back to 500 when I left.

It seems like items in general don't get stored in your inventory. When I pick up a sword, it disappears completely, it didn't appear in my inventory it was just gone. It made fighting the boss pretty difficult.

Also you can't die in the game.

The buttons and shops do not work, my health would drop but my character would never die, pressing escape while in a dialog menu will bug out the camera, and I think that is about it.

You can jump on enemy heads to avoid them. Enemies walk through walls. Got stuck on a merchant's screen, talked to them and couldn't press the button to continue away from them (no cursor). Other times I had the cursor on my screen the entire time playing. Didn't seem to be an end to the dungeon. Even a placeholder portal at the end to bring you back would be nice. I couldn't hit anything with my swords. No amount of head smashing keyboards helped. I got my first weapon half way through the first dungeon. Had to reprise my role as keep-away legend until I came across one.

Some enemies were able to teleport through walls. Walking sideways causes character model to slide.

Post-Mortem

Amana Color

What Went Well:

- **Documenting Process = Easy Iterations**
 - Documenting our process helped us understand what we were missing and what elements we could add into the game that would be high priority for the player. Since we documented from the beginning, it was easy to pin-point and eliminate elements that wouldn't be adding to the player's experience.
Some elements that were added to enhance the player's experience were particle effects, sounds, and hit feedback animations. We realized these were important because if we added all the elements we initially wanted, they wouldn't have been as enjoyable for the player, thus, making the game not desirable to anyone who would have to play it.
- **Team Communication**
 - During the production of our game, our team got together frequently to discuss Jira, Confluence and Weekly scrums. Having the open communication, we did led to mutual understanding on anything that was new, being worked on, or difficult to implement in the game. Since we had the limitation of the pandemic, talking frequently made it easy to keep on the ball and understand what parts of our premise we wanted to keep working on.

What Went Wrong:

- **Working from Home; A New Adventure for All of Us**
 - Unfortunately, working from home was difficult for most of us. Having the culture of being able to be productive around your team-mates and have someone point on a screen when something went wrong is something we all took for granted when we encountered the stay at home order.
Since most of us have different home situations, sometimes it was difficult to stay motivated and keep a steady workflow. Not having a quiet space, or access to the hardware we needed, hindered us in the beginning. Some of the issues we dealt with hurt our workflow and production time. If we were in our normal location, with the technology we needed to complete the game and the human connection we all desired, I believe MythGrove: Festival of Yew would of been further than where it is now.
There is a sense of accountability when seeing each other in person that I, truly, felt left as soon as we were told to stay home.
- **Senioritis**
 - Oh yes, it's the *MAD SENIORITIS* that we all feel when we get closer to the end of the school year. Since our project was a cap stone project for school, and during the final countdown to our degree, we all felt done by the second quarter of this project. Being exhausted from having the distraction of other classes hurt our chances of increasing our workflow for the final part of iterating our project.
- **Overriding in Git**
 - The team was constantly overriding each other's work through git. This caused a lot of issues and extra time needed on the project in order to have a stable build.

Lessons Learned:

- **Networking is a Pain**
 - Unfortunately, networking was a pain. When establishing networking you must write the code so that it works with networking. When you code a network game, everything must be coded with the thought that networking will be implemented in the future. Then you must use the networking syntax, that sends everything to the server and back to the player. Something as simple as jumping has to be more thought out because you are not receiving the jump animation feedback on one end, but multiple players' end. We did not fully plan out what was going to be networked which is why we had so many issues establishing networking to our game. Without the background thought of "this might need to be networked", we had to rewrite a lot of scripts that would in return break other scripts and our game. We had to code everything that would target all the clients rather than target locally. So, coding things to be over the network took longer, gave us a lot of bugs and opened us to many technical limitations because the software mirror does not like nested networkbehaviors which made us have to incorporate mechanics at runtime.
It also did not help all 3 programmers did not have previous knowledge of mirror, which hindered our workflow only because they had to learn a whole new software and how it functioned with our desired mechanics.
What did help is that they increased their knowledge of the software mirror and how to incorporate networking into other future games.

Jason Skillman

What went well

- The planning process went very well and was completely documented. This helped create a strong foundation for the rest of the team to build off of during the development of the project.
- The team was helpful in helping each other out when they needed it. We constantly were helping each other with design and technical issue throughout the entire process.
- The game was very close to the concept art and stuck with the lore of the game.

What went wrong

- Team was unable to help each other directly/physically. This stunted the growth of the entire team and the product itself.
- Networking the gameplay was a huge undertaking. Was unable to get some simple gameplay mechanics functionally working. Lost lots of hours trying to debug networking issues.
- Team did not have an animator so animations took four times as long as it normally should have.
- Team was constantly overriding each other's work through git.

Lessons learned

- I learned a lot about how networking gameplay works.
- I spent too much time debugging issues instead of adding new gameplay. I need to learn when to cut it off.
- Learned how to effectively work remotely as a project manager.

Joel Hanson

What went well:

- **Communication**
 - During this hard time, and staying home, communication was key in order to make this project succeed. And for the most part, we communicated everyday. We had 3 scrum meetings a day, and everyone would talk about what they were doing and what their next plans were. It was a huge process that I believe we nailed as a team
- **Documentation**
 - We easily documented everything that was going to be in the game. everything was tasked out to people, so everyone knew what they were to work on. If there was something that needed to be worked on, a task was made. Everyone was up to date, and everyday Jira was update

What Went Wrong:

- **Networking**
 - This was a huge undertaking, and I believe one of our biggest downfalls in the development of the game. There was countless hours spent trying to put in all the mechanics into the game while trying to get it to network well, especially with little to no experience in something like this. Many hours were spent debugging bugs or trying to figure out why one thing works but the other doesn't, etc. It was a huge pain.
- **Corona:**
 - I believe this was the biggest problem that was wrong. It was hard to help someone if they had tech issues or something. So that person was on their own if it came down to it, the most someone could do was try and help over a call. Also the transition to online learning was also a huge undertaking and getting used to.

Lessons Learned:

- Networking is hard
- I learned a lot how networking works though
- Even if you have a really really great team, things can still go wrong, which is okay.
- Corona is awful

Isabella Weikman

What went well:

- We all stayed pretty comfortable with each other as teammates and I feel that we worked well together because of this.
- The game's aesthetic matched the concept art pretty closely, and I believe that this happened because I provided a lot of references and in depth concepts really early on. Keeping people on track with even just a color palette can really help a game look visually appealing.
- From the beginning we put a lot of emphasis on maintaining an organized documentation process so that we could keep track of things that need work, and iterate in a way that gave us a safety net if things went wrong.

What went wrong:

- We lost steam about half way through the production process, and motivations were low. I feel like most of us got sick of working on the game for a myriad of personal reasons. Life got crazy.
- 7 people is a big crew, and that could have been a factor in people not knowing what to do next, since delegating tasks throughout the team often lead to someone having too much to do and someone else having too little.
- We were so unaware of how the shops were going to work until the final weeks, and even at that point there was still confusion. If we had planned more in-depth in the beginning, the shops could have been more fleshed out by the end.

Lessons learned:

- Be honest with your teammates. Say when you are available and when you are not. Don't include something subpar and out of place for the sake of saving face. Communication communication, and don't let hurt feelings ruin a game.
- Scrum is essential, and keeping track of individual tasks (while tedious) is most helpful in the long run, especially when tracking progress.
- Give everything a reason. Don't try to add something in under the impression that you can fix it up and flesh it out in the future. Make decisions earlier so that executing is faster.

Juan Santos

What went well

- I felt for the most part my team did a great job of staying communicated and transparent, often there would be a post of what was being worked on by individuals members and I believe that feed it to motivating the rest of the team.
- Documentation was very well managed and overall allowed the team to stay very organized.
- Our preproduction was done before the class started and I feel because of that we had a good idea of what needed to be done as well as having everyone be on the same page.
- I felt the combo workflow of Bell doing the concept art and me doing the models allowed for some great assets, I was very happy with the overall look of the game.

What went wrong

- After the first quarter things slowed down quite a bit I believe this is because of the amount of time people spent working on the game in the first quarter, the team was just tired after the initial burst.
- I was in charge of animations
- I heard that networking slowed the progress of the game's coding considerably
- Though I think the scope we set was still achievable, the way we went about building the game wasn't the most optimal.
- Not having face to face contact with people I felt really effected the workflow we planned on having, it also made communication less intuitive.

Lessons learned

- How to communicate with my team as well as making sure my tasks on jira were being updated
- How to use many great software, like substance designer, motion builder, and blender
- To work more efficiently, I got good at pumping out models.
- Push your team mates to do more or try new things
- A good concept artist is essential to keeping a consistent look to the game.

Kyle Gray

What went well:

- Learning - In the end, this senior project was just another class which is meant for us to learn. I'll go over some specific things that we learned later, but using new tools and working in a diverse team allowed us to grow as game designers and developers.
- Version control - Using Git worked great for us. We had to deal with the occasional merge conflict, but being able to work in branches was great for our productivity, if we make breaking changes then other people can still work on things while we fix them.
- Communication - While we were working on the project we spent a lot of time in meetings, we talked a lot through messages in Discord, pretty much whenever we needed something somebody was available to hop in a chat. Communicating well was a big part of what went right in this project, if we weren't all on the same page all the time there would be a lot of backtracking to get everyone on the same page and to change features that somebody implemented that wasn't supposed to be, and if we weren't available to help each other it would have taken us a lot longer to complete our tasks, and we wouldn't have had as much done.

What went wrong:

- Playtesting - I wish we had done a lot more playtesting and gotten a lot more feedback. In the article I linked about Cultist Simulator, the developers used a strategy called open production. With open production you are constantly taking in feedback from the community and giving out small updates frequently to test new features and get more feedback. We only did a few playtests and only really discussed the results of one or two, and if we had gotten more feedback we could have made a game that is more fun to more people.
- Networking - Networking ended up eating a lot of our time, especially at the end, and not very much came from it in the project. Of course we all still learned a lot about networking, we would likely have a much easier time next time implementing it. However, this time it could have had a lot of improvement in the final product.
- Scope - We originally had a lot planned for the final game, multiple dungeons, tons of enemies, tons of modifiers, but it was a lot more difficult adding all that content than we thought, it took a lot of time to get the core functionality of all these systems in place we hardly had time to fill them with the content we planned.

Lessons learned:

- If you're making a multiplayer game, make it multiplayer all the way through - Our strategy for making the multiplayer in this game was essentially to make it work in singleplayer first, then add networking later. Turns out, this isn't a very good way to do networking. It wasted a lot of time redoing the same code when we could have had networking in mind when we made the code originally and had it already network ready when it was done. If we did this I think we would have had a much better game in the end, one which at every stage could be played with the whole team or playtesters.

- Playtest often - The feedback from playtesters is super valuable. Not only will playtesters find bugs and break the game in ways you never thought of, but they also will show you where the fun is in the game. If I were to do this class again I would focus on getting the game loop done within the first few weeks, and playtest every week after that incorporating the feedback throughout the process.

Christian Martin

What went well:

- Team communication was abundant as the virtual environment we have been subjected to has lead to constant talking and working while voice-chatting
- Documenting the process as we went helped to define goals and set the pace for the game
- Each week tasks were distributed and a goal was set for the week on aspects of the game that were to be done. Alternating priorities and having tasks be distributed as evenly as possible among teammates helped to complete as much as possible for the weekly sprints.

What went wrong.

- Overscoping. A lot of plans in the beginning fell through due to time constraints and other networking and bug issues taking up a lot of time.
- Working from home is a big challenge especially when we don't have that interpersonal connection between teams. Spending long amounts of time communicating on a screen is incredibly tiresome.

Lessons Learned:

- I personally could've stepped outside my comfort zone more. I learned a lot more about modeling, and how I need to be able to wear more hats in a small team rather than just being directed to one position.

Technical Limitations

- Mirror does not like nested “NetworkBehaviors”, so certain elements to game play could not be placed in elements that had “NetworkBehaviors” already there, so it had to be done at runtime.
- The only computer the game can be played on is a windows computer. The file does not work on Mac and Linux, so it is then not available to all players.
- Network bugs were very difficult to test and debug.

Credits

Programmers

Jason Skillman

- Project Manager
- Dungeon Programmer
- UI Programmer
- Tools Programmer
- Level Design

Joel Hanson

- Lead Programmer

Kyle Gray

- Lead Networking

Designers

Amanda Coler

- Co-project Manager
- Lead UI/UX
- Sound Designer

Bella Weikman

- Lead Designer
- Lead Character Artist
- Lead Sound Designer

Juan Santos

- Lead Modeler
- Lead Sculptor

Christian Martin

- Modeler
- Narrative story writer

Special Thanks

Jeff Knapp

- Music Composition

[Link](#)

Itch.io URL: <https://jason-skillman.itch.io/mythgrove>