# Gata Guressi

*Appendix A includes a dictionary of fictional terms, which may be helpful when reading this report.*

# Narrative/Game Context

## Overview


Quuvol is not trusted

*Gata Guressi* is a sci-fi exploration RPG tech demo. You play as Quuvol, a merchant from the planet of Guressen, who is visiting Kallay Tirridor on the planet of Voenn to sell ielsek, a valuable resource. Guressen's immoral former government was feared for practising slavery, leading the rest of the system to fear all Guressi (Guressen natives). While attempting to sell ielsek, Quuvol faces characters who curse him and refuse to trade because they believe he is a slaver.

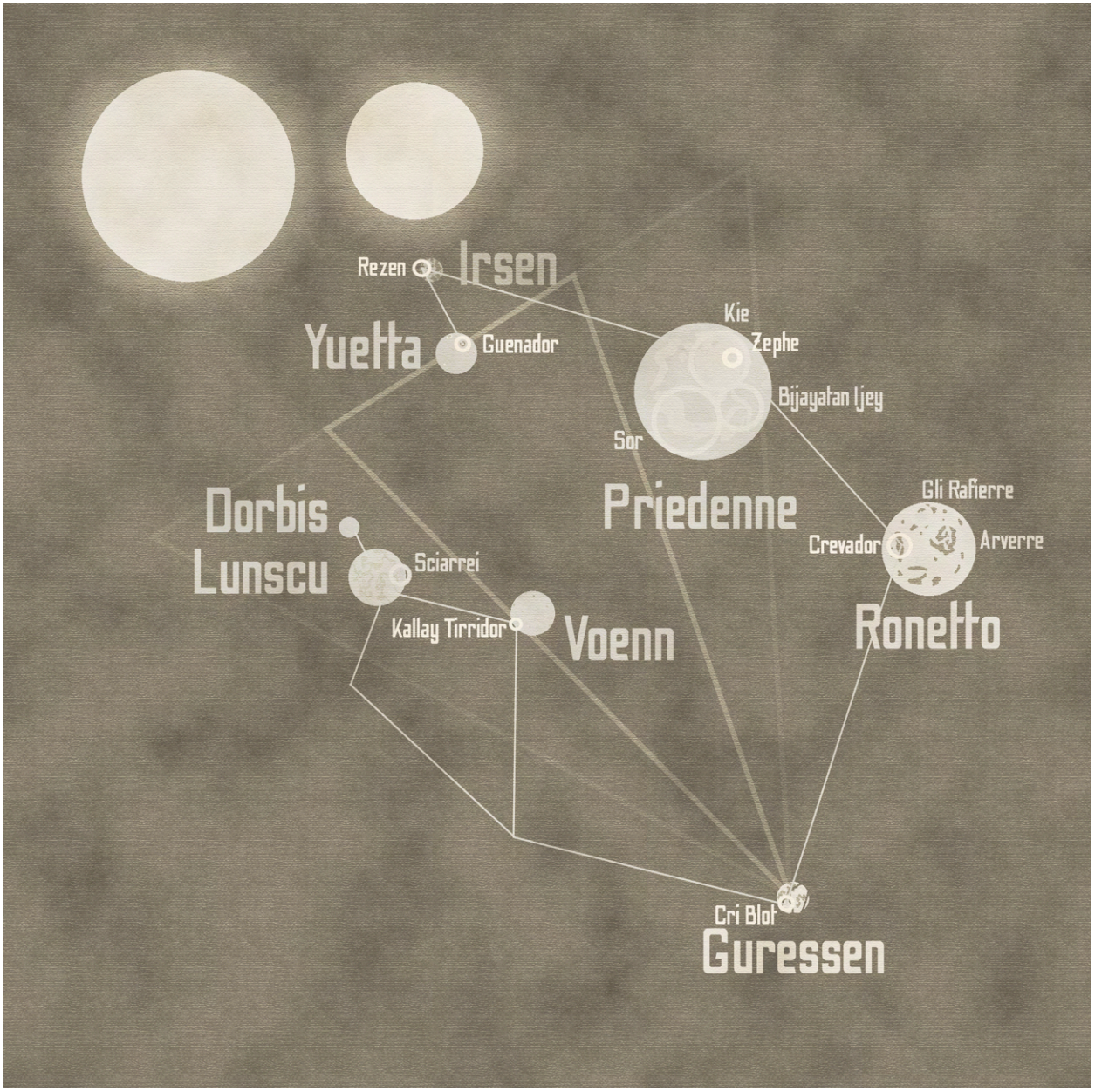
## Project Aims

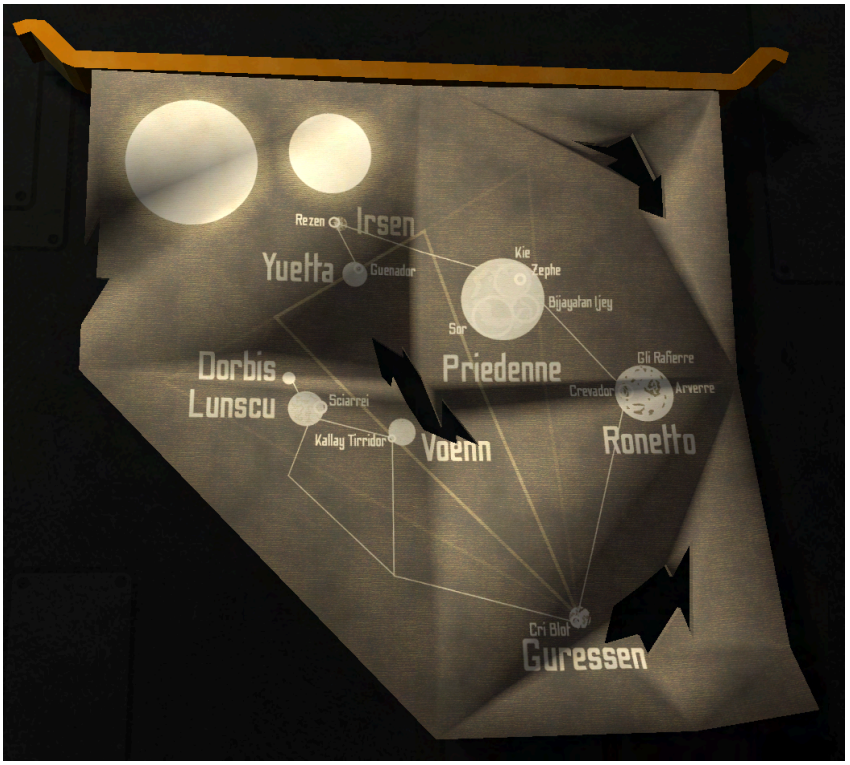
Entrance

I aimed to develop a polished, engaging experience that simulated a slice of a larger sci-fi RPG. Cohesion, immersion, and optimization were my main priorities. To create a cohesive, immersive environment, I would allude to a larger sci-fi universe through the environment and dialogue with randomly generated characters. These characters' activities, cultures, and prejudices would bring life to the environment, building immersion. Roleplay opportunities and small, optional interactions would also boost immersion. The player would arrive and leave through cutscenes, creating a sense that the world exists outside this game’s scope. I aimed to optimise the game for the lab machines by using baked lighting, the built-in RP, a custom master shader, low-poly modelling, occlusion culling, and an area loading system.

# Experience Description

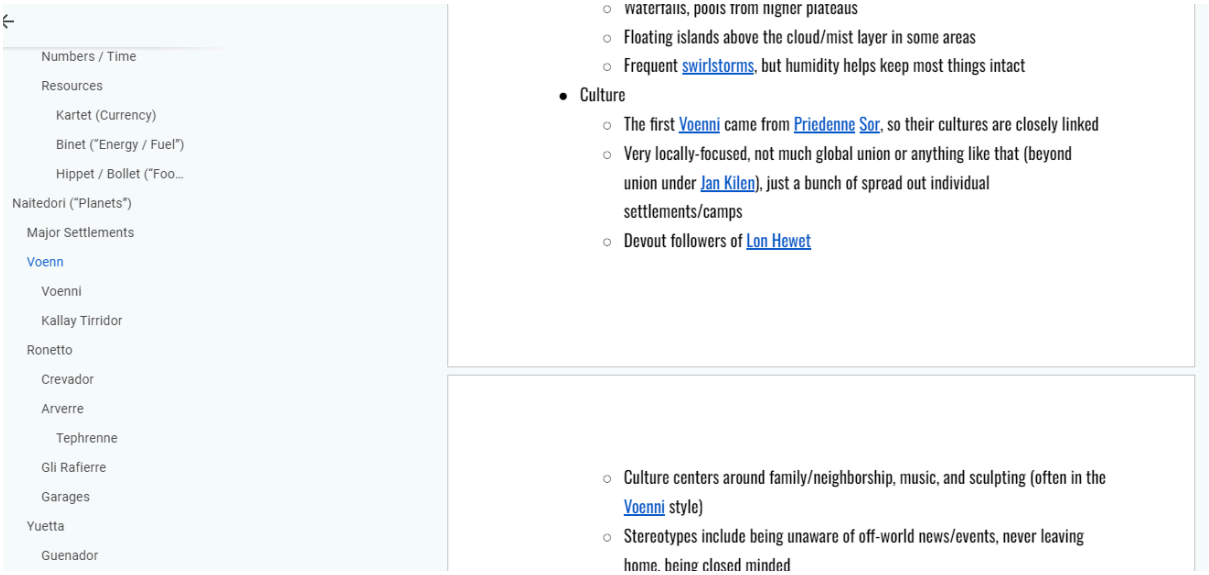
## World (The Hierre System)



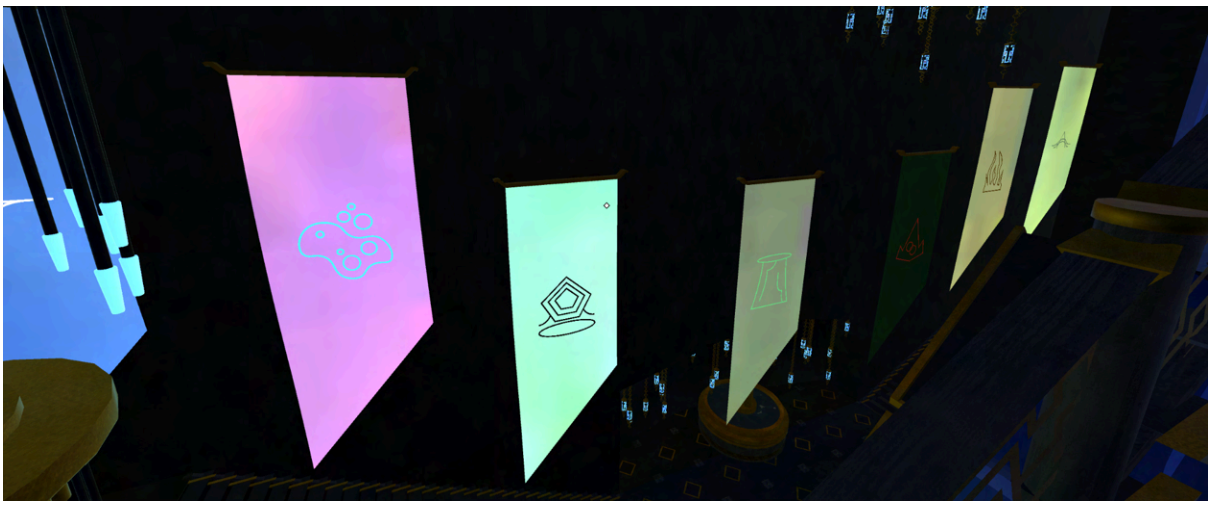



Map of Hierre's planets, major settlements/regions, and trade routes

The Hierre system consists of 8 planets divided into two regions: the northeast region, Hierre Kie (containing planets Yuetta, Irsen, Priedenne, and Ronetto), and the southwest region, Hierre Sor (containing planets Guressen, Voenn, Lunscu, and Dorbis). *Gata Guressi* takes place on Voenn.

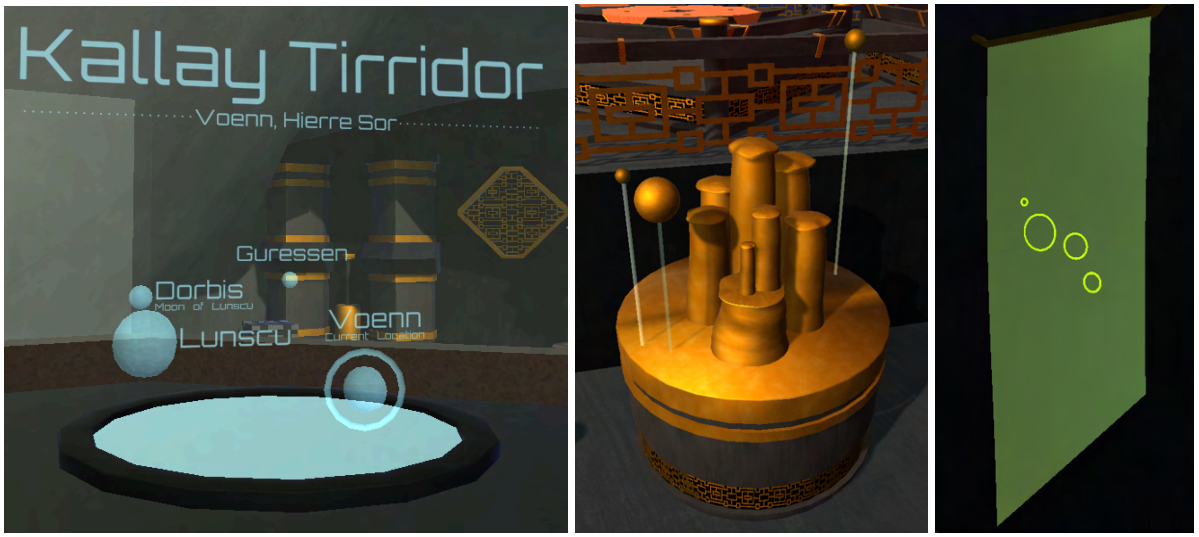



My 100+ page Hierre document



Planetary banners

Each planet/region has an icon, culture, aesthetic identity, and language. You can learn about each planet/region by speaking to its inhabitants and inspecting cultural objects like banners.




Hierre Sor representations

*Gata Guressi* takes place on Voenn, but nearby planets are visible in-game in the skybox and space flight outro cutscene. They are also represented on your spaceship's projection map and in sculptures around Kallay.

## Gameplay


Cutscene

The game begins and ends with cinematic, scored cutscenes depicting Quuvol arriving at and departing from Kallay. These immerse the player in the world, leaving the impression that Quuvol's life will continue past the end screen.


Selling ielsek

The player's primary goal while visiting Kallay is to speak to others like a door-to-door salesman in hopes they want to trade for ielsek. This adds an overarching goal and explains Quuvol's motives. However, selling ielsek takes a backseat to learning about the setting through the dialogue and environment.


Translator

Quuvol uses a translator, which must be tuned to the correct language to decipher character dialogue and text. While most characters speak pure Hiesca, the common language, some use words from (or speak entirely in) their native language.


Dialogue with context

The dialogue system was designed to help the player actually learn these languages if they wish. Some recurring words (e.g. “iel”, an abbreviation for “ielsek”) become cumbersome to translate individually, so the player will learn what they mean through exposure without needing to use their translator. Some phrases (e.g. without direct English translation) also display additional context.


**Lunsca**

- Lunscu and Dorbis planetary language
  - Very recognizable but hard to understand phonetically to non-fluent speakers
  - To everyone else, it sounds like "isuceiuiseiceuscusiscsieuacei"
- Conventions
  - Pluralized with *-es*, not *-i/-en* (like English)
- Words
  - Nouns
    - *Sci* = the viscous, bubbling pink substance found all across Lunscu
    - *Ki* = I, me
    - *Kuscied* = friend, bro, buddy
    - *Usciecu* = tattoo. Tattoos have big cultural importance in Lunscu
    - *Asca* = joy, happiness
    - *Tucas* = game

Languages sound distinct

Each language has conventions and an auditory identity to build cohesion. Lunsca is nearly impossible for non-native speakers to understand. Sorpri uses long, flowery words and names instead of pronouns. Hiesca uses consistent prefixes, suffixes, and sounds to build its identity (e.g. the sound "*ie*", two-syllable words, locations of importance ending with *-enne*, *-erre*, or *-ette*).


Backstabbed!

Immersive roleplay elements include watching a band play, spectating a boxing match, sitting with other NPCs, and buying drinks. Smaller, optional interactions further build immersion: a hardcore card gamer will get upset if you jump on the table when he has almost won a game, and bartenders do not appreciate you stealing glasses from behind the counter. Quuvol will express his thoughts upon inspecting various items aboard his ship.



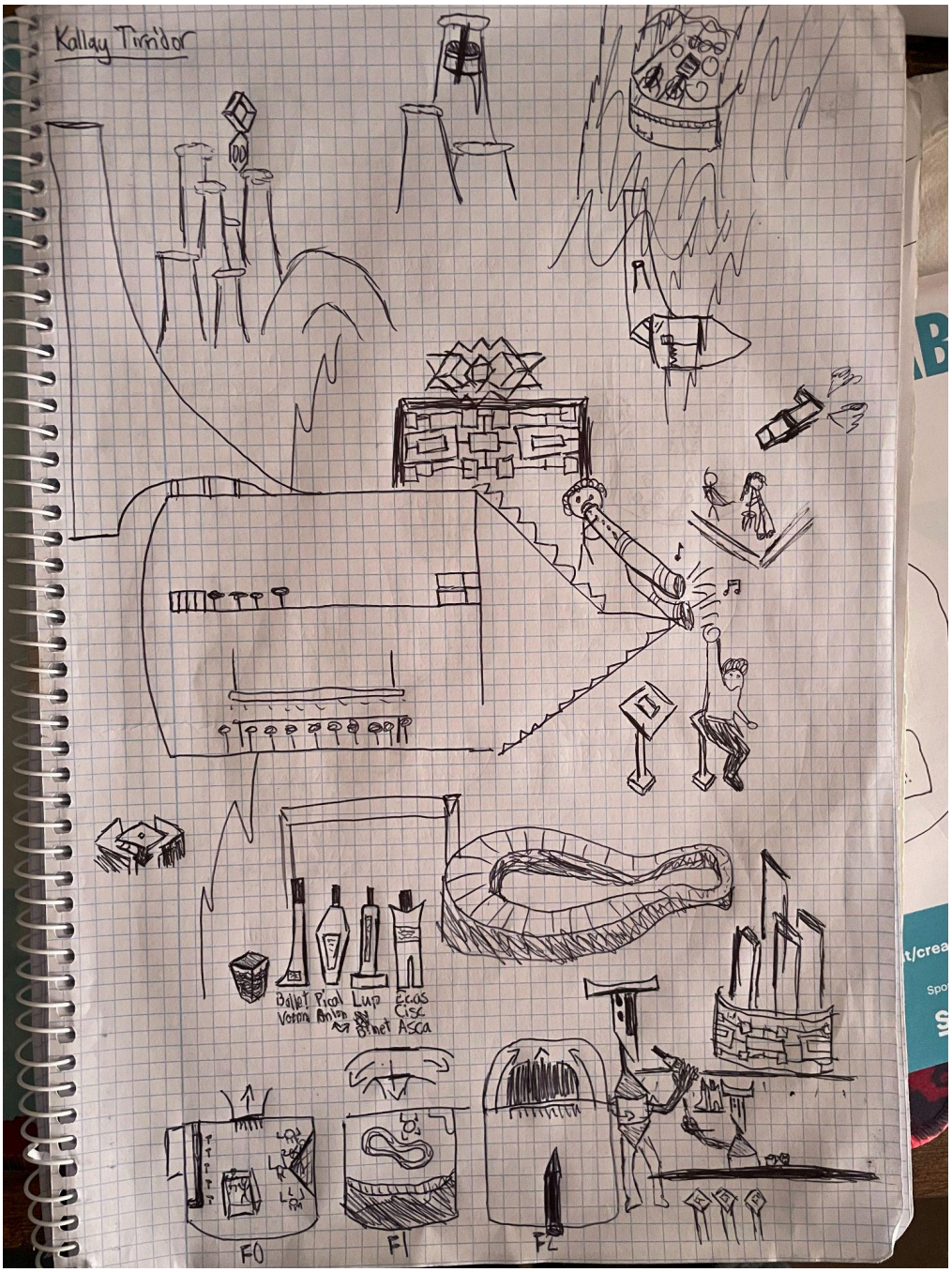

Easter eggs

There are also secrets and easter eggs to be discovered.

# Aesthetic Rationale

The game was visually inspired by *No Man's Sky*, *Cyberpunk 2077*, and *Star Wars: Galaxy's Edge*. Its soundtrack draws on the synths and processing of *Hyper Light Drifter* to build an exotic sci-fi feel.

## Kallay Tirridor




Concepts

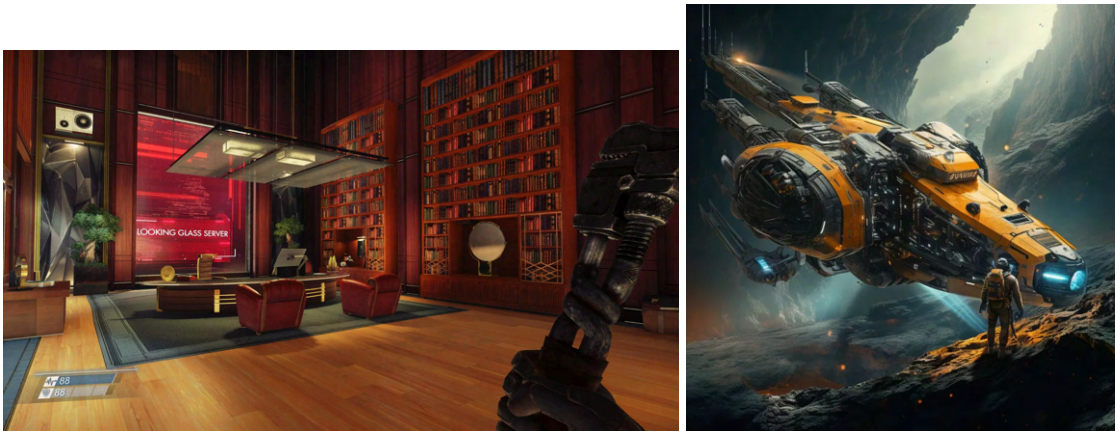

*Prey*'s Talos I Lobby (top) and *Cyberpunk 2077*'s Heavy Hearts Club (bottom)

Kallay's biggest inspiration was *Prey*'s Talos I lobby and its warm, geometric aesthetic. I aimed to create a lively, inviting, exotic location where cultures collide (inspired by *Star Wars*' Cantina) with characters engaging in varied activities. I used geometric designs and LED lighting to create a futuristic atmosphere.


Statues (Braavos, Kallay)

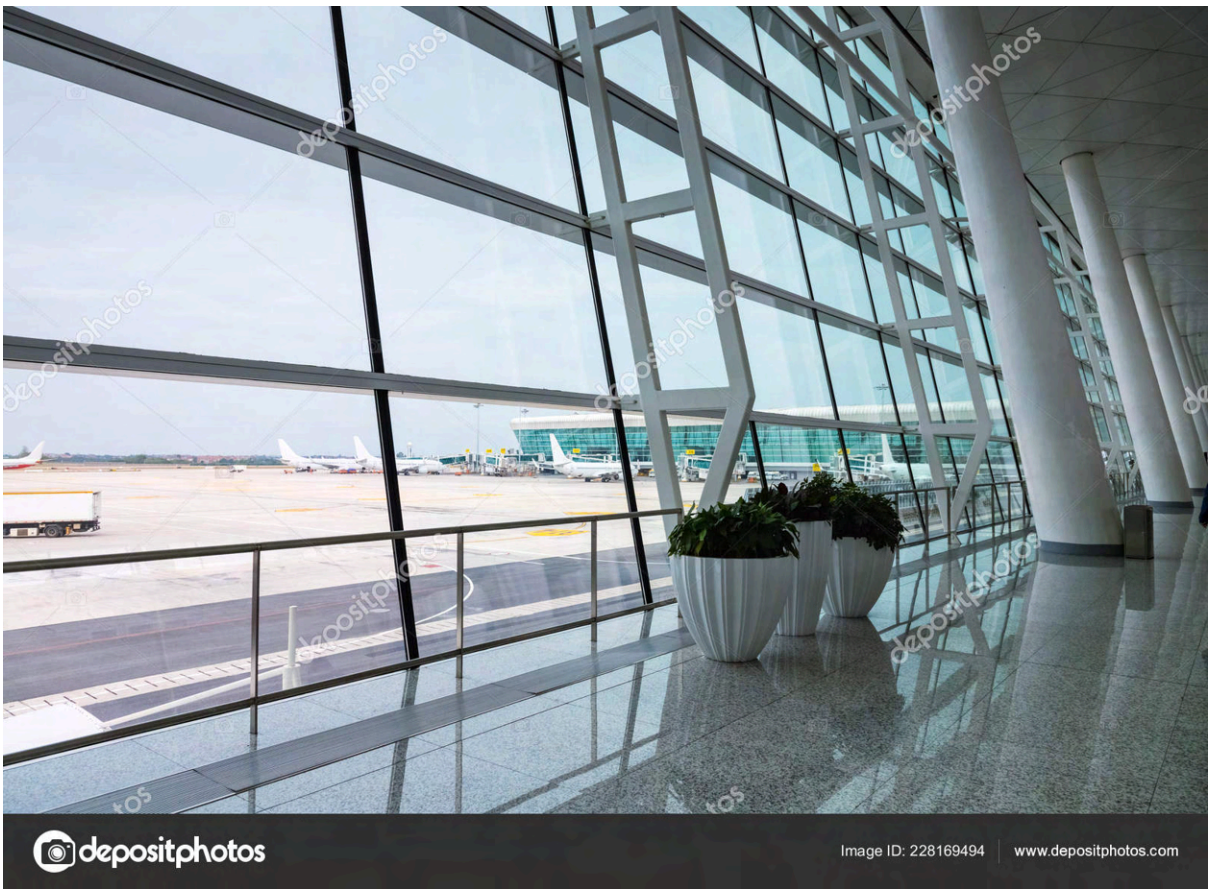
Gold, statues, and a grand staircase entrance imply the bar's wealthy target demographic. The authority and grandeur of *Game of Thrones*' Braavos statue inspired the statues outside the main entrance.

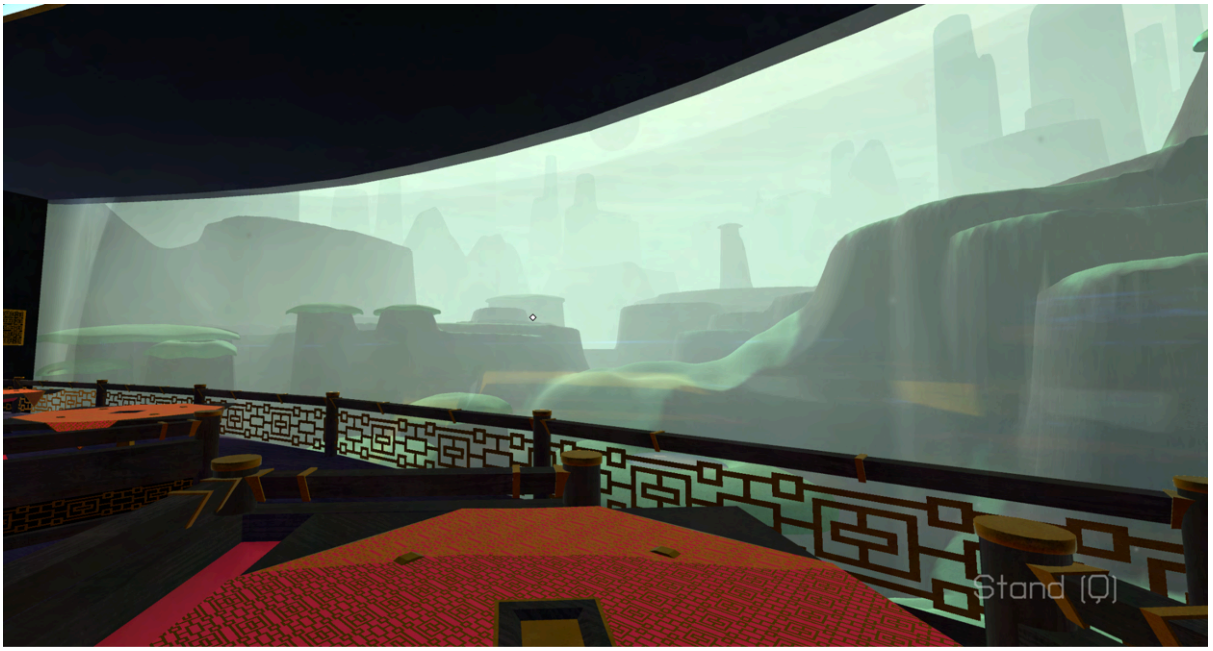
Initial colour palette




Desired (left) vs undesired (right) colour palettes

I added hanging cloth, misaligned barstools, animated NPCs, and a warm red-gold-black colour palette to make the space feel lived in. I drew inspiration from traditional Indian designs and lamps for a culturally rich appearance. Kallay's red contrasts with Voenn's alien green, making Kallay feel safer than outdoors. The colour palette was essential for making the bar feel inviting; I avoided a stereotypical sci-fi white-grey-blue-orange palette as it felt cold and metallic.

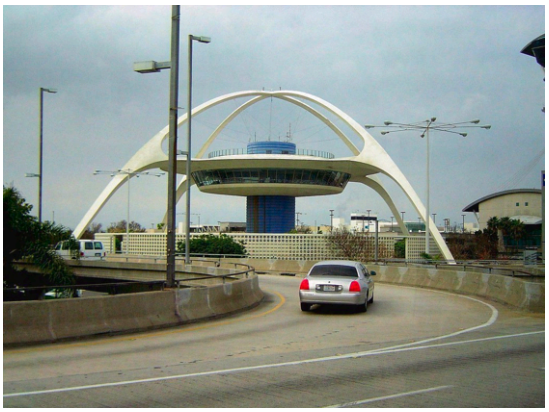



Windows (airport, Kallay)

A huge window inspired by airport windows opens up the space. Like an airport, visitors could watch departures from here. It also offers a luxurious view to the upscale loft seating and bar on Floor 1.

## Bars

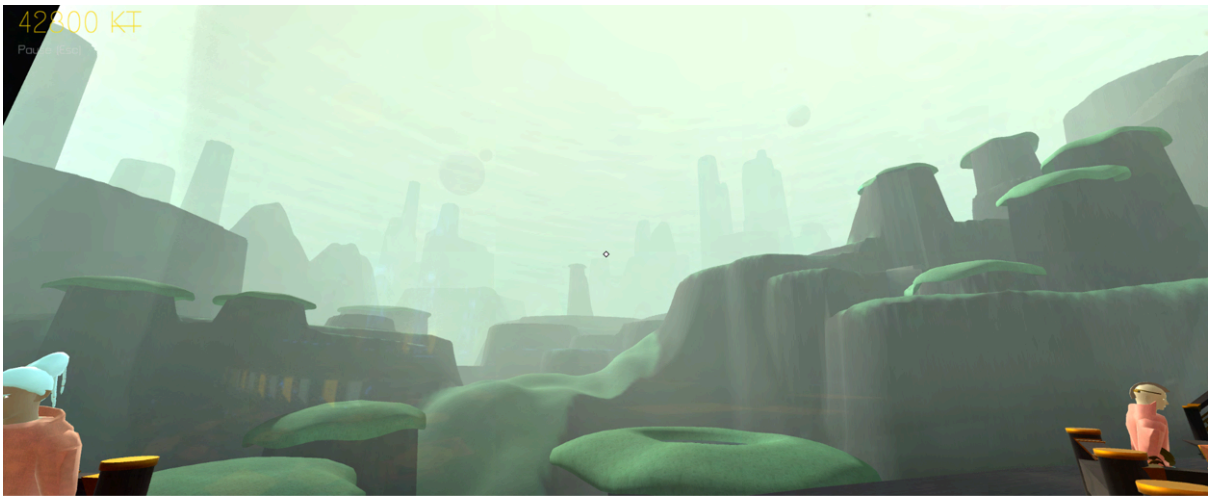





Inspirations

The conveyor bar on Floor 1 was inspired by the Space Age aesthetic, rotation sushi restaurants, and a rounded cafe bar at Eataly in London. Other inspirations include *Cyberpunk 2077*'s Heavy Hearts Club and *Star Wars*' Cantina for their futuristic bar designs.

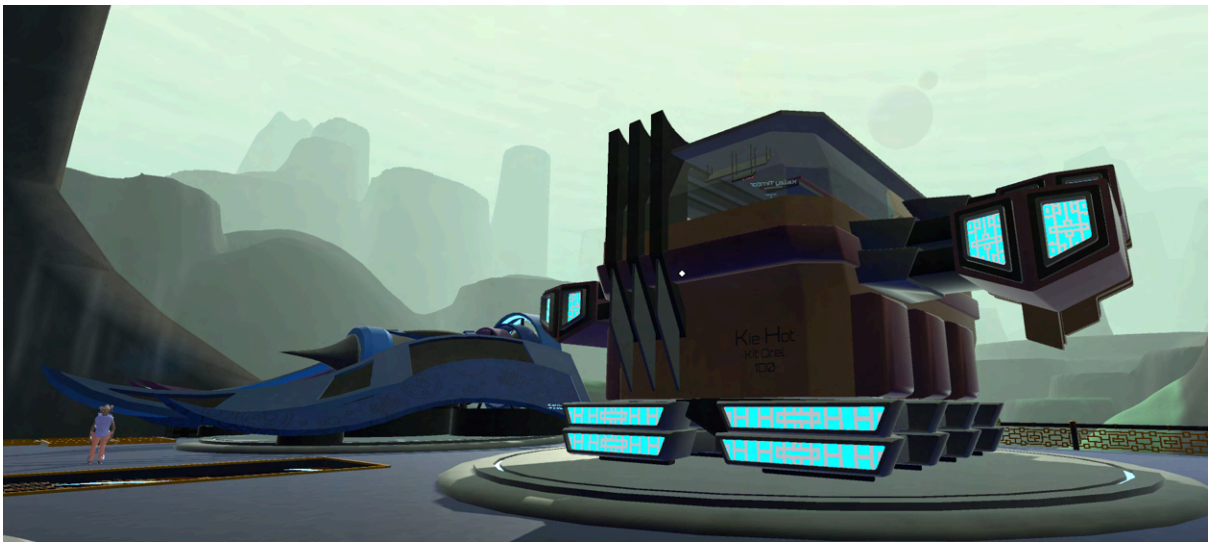
## Voenn



Rock pillars

Red Rock Valley inspired Voenn's rock formations. Rather than leaving these pillars barren, I added mossy grass hanging over the edges to enhance Unity's heightmap landscape. The skybox is cloudy and matches the fog colour, making distant pillars fade into the mist. This communicates Voenn's uncivilised, mysterious nature.

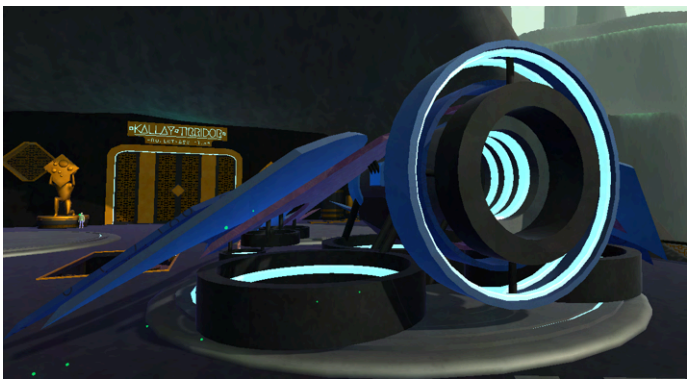
## Spaceships



Upper-class landing dock

The two spaceships were designed to contrast dramatically, showing the breadth of how interplanetary transit looks in Hierre and, at first glance, implying that Quuvol could be earning more. Quuvol is out of place in the high-class ship dock, alienating the player and their bulky transport ship.


15
42800 KT
Pause (Esc)
Translating:
HIESCA
(drag slider to change language)
- Glim Pilanon -
Glott hioh 1E2 Hal. Weg pa hocal gathal criti kartet keiutah tih.
It's a 482 Hal. One of the best money can buy.
Continue (E)



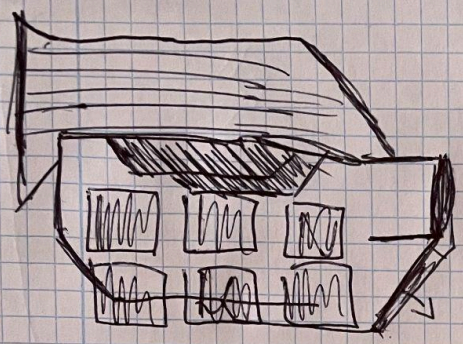
Hal

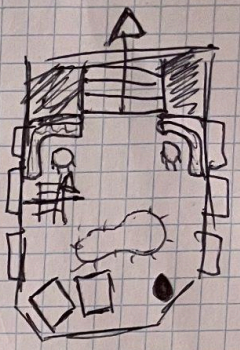
The Hal is an exotic, curved, stylish ship popular with younger generations. I aimed for expensive flair and a sharp, edgy appearance similar to a wasp or needle.

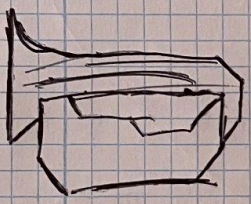
*No Man's Sky* fighter, *Monster Hunter* armour

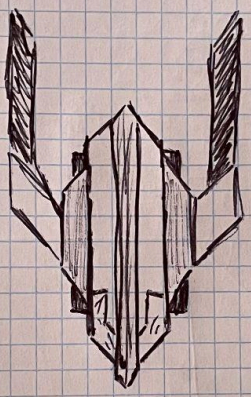
Syringes inspired the ship's pointed body. I drew on fighters from *No Man's Sky* for the rings around the engine and landing gear. The wings cover the ship's body like a faux fur scarf; a *Monster Hunter: World* armour set's collar became my primary reference point. A subtle colour-changing paintjob solidifies the ship's style-over-substance mentality. The ship's structure accentuates this: the cockpit is much smaller than the player's, meaning the pilot would be cramped during flights. This ship is designed to sit and look pretty.

Kit Orel



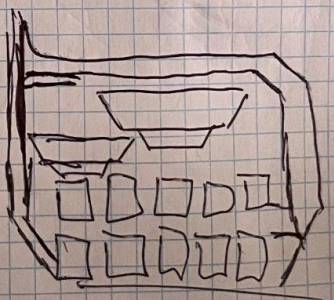


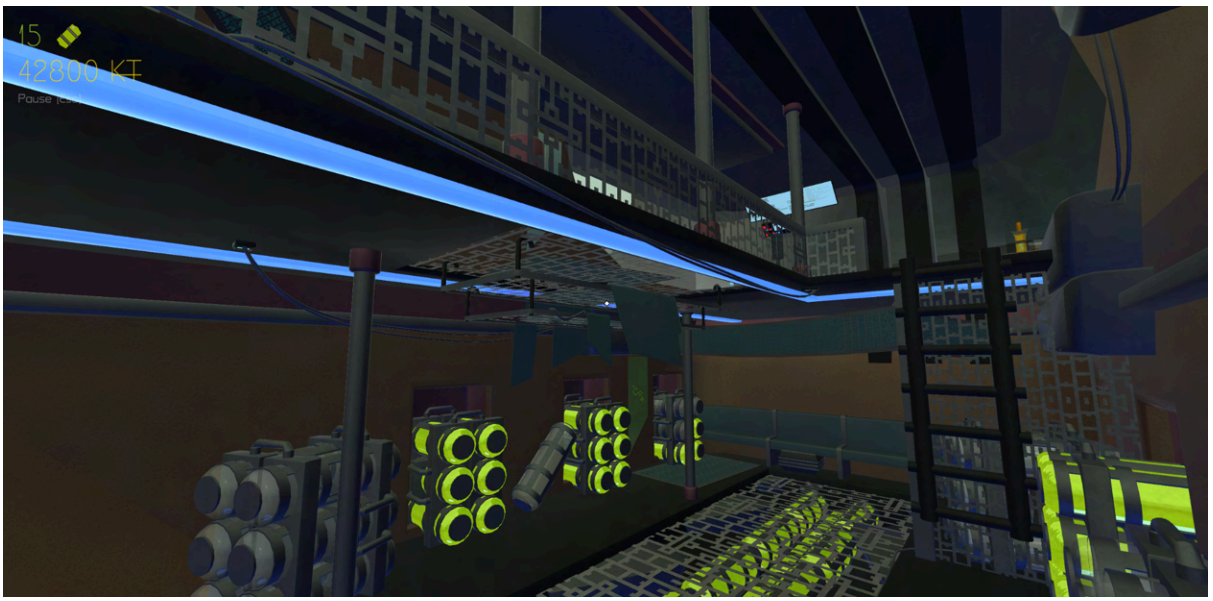




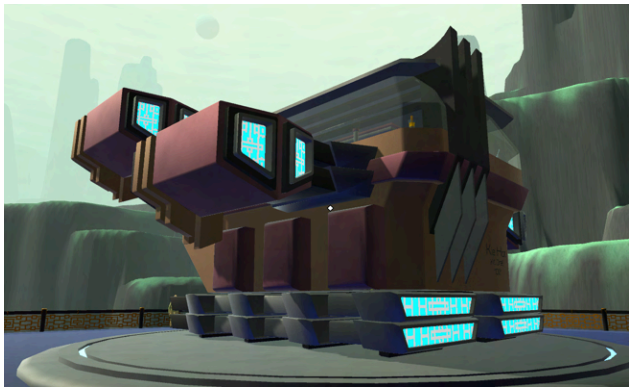
F1

F0









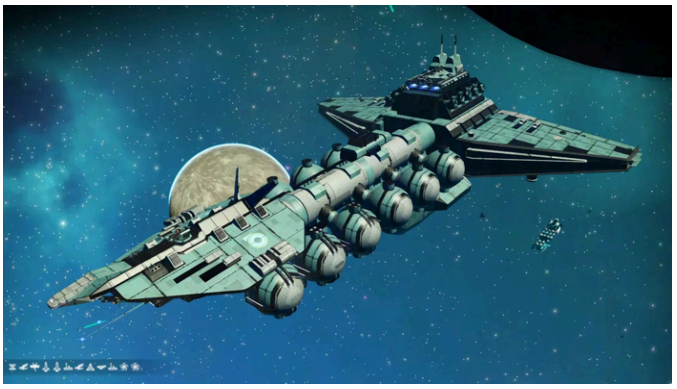
Kit Orel

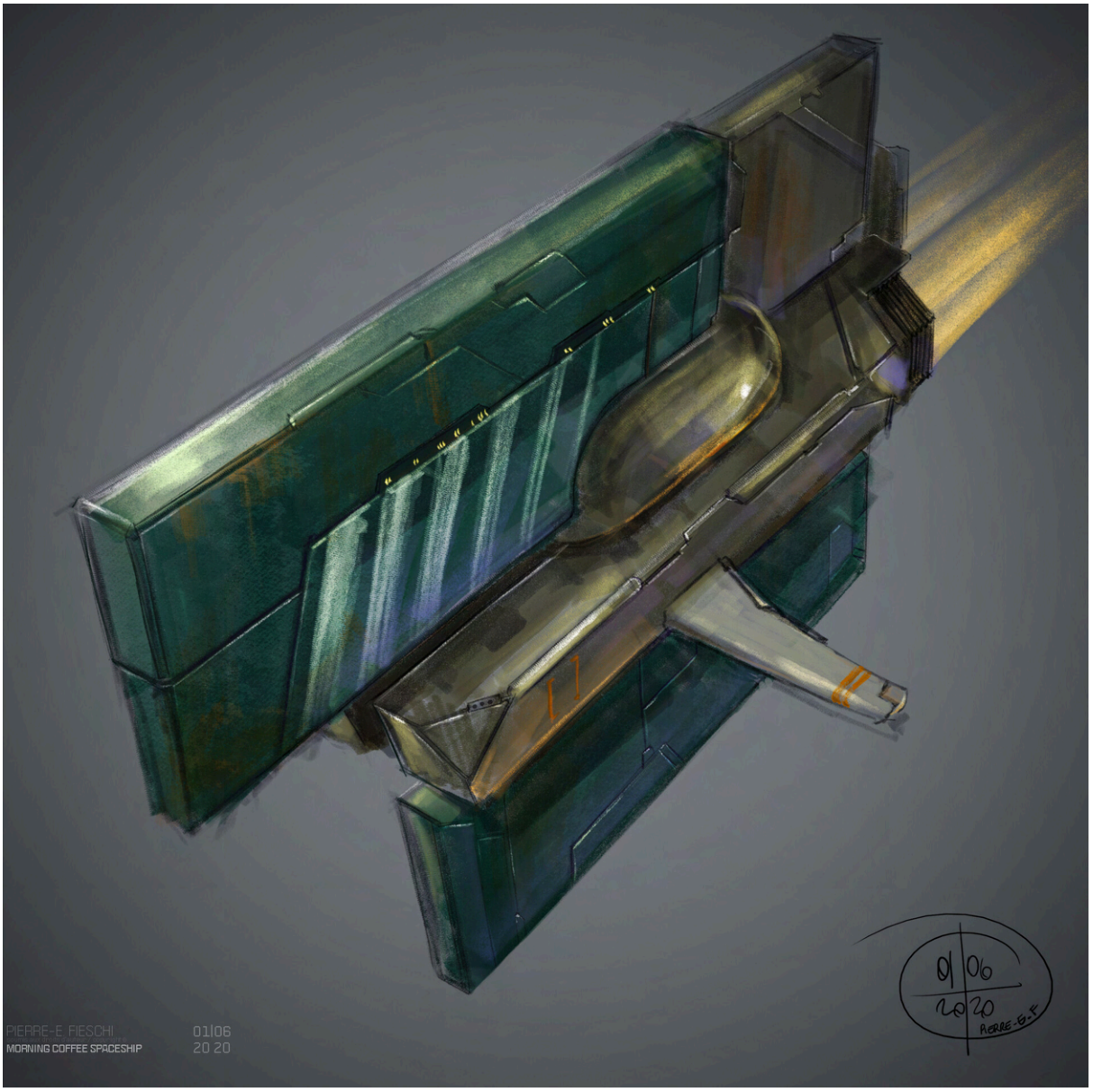
The Kit Orel, the player's blocky merchant ship, is the interplanetary equivalent of a truck. Designing this ship around its interior was beneficial for a utilitarian appearance, as the space needed inside dictated its exterior.



Interior inspirations

The interior was inspired by *Cyberpunk 2077*'s crowded Japantown apartment and ripperdoc offices and *Star Wars*' Millenium Falcon. The interior includes a bed, couch, drinks, and other details showing that Quuvol lives there during long flights.






Exterior inspirations

The unforgiving, blunt appearance of garbage trucks inspired a rough-hewn exterior. I also included exterior storage pods inspired by *No Man's Sky*'s freighters.

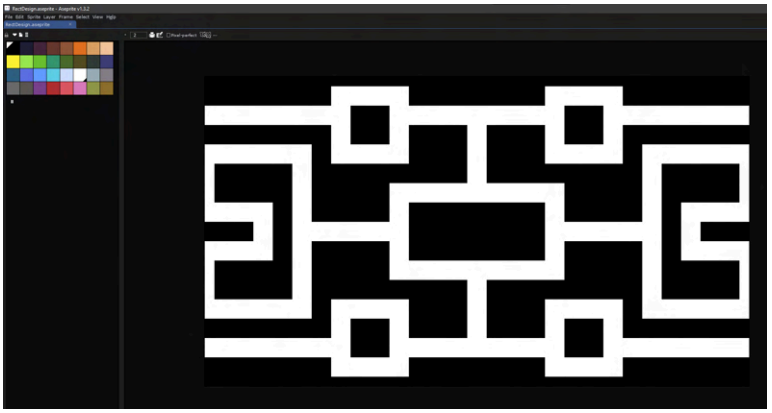
## Clothing

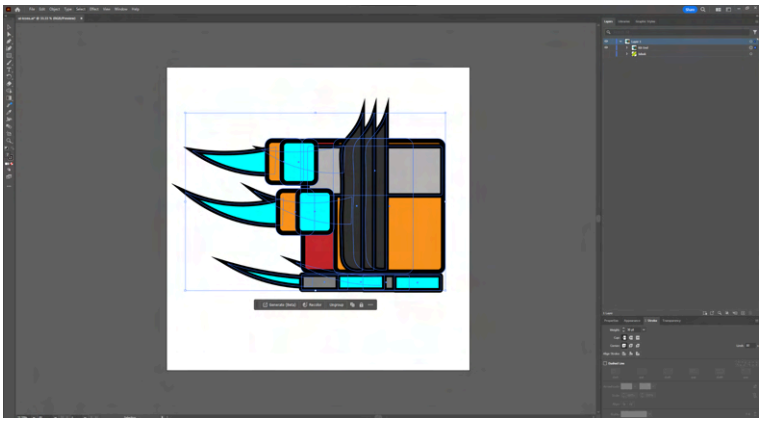

Upscale merchant fashion

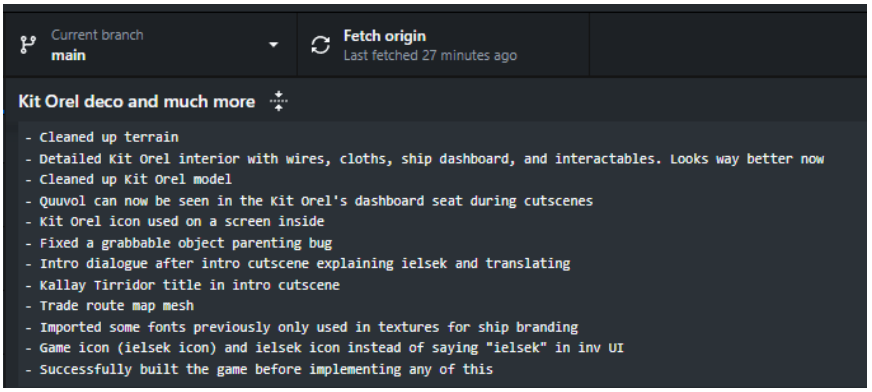
High collars and asymmetric sleeves are in fashion, inspired by *Cyberpunk 2077* and the Met Gala. High collars accentuate merchants' individualistic, sceptical attitudes. Abrupt, spiky protrusions from clothing belong to the fictional upper-class Voenni aesthetic. Clothes feature metal and LED ornamentation, representing technology’s popularity in fashion and body implants.

# Technical Implementation

## Software








Aseprite, Illustrator, GitHub

I created sprites/textures using Aseprite, Illustrator, Photoshop, and GIMP's Tile Seamless filter. I produced foley and synthesized soundtracks/SFX in Logic Pro and used GitHub and Trello for project management.

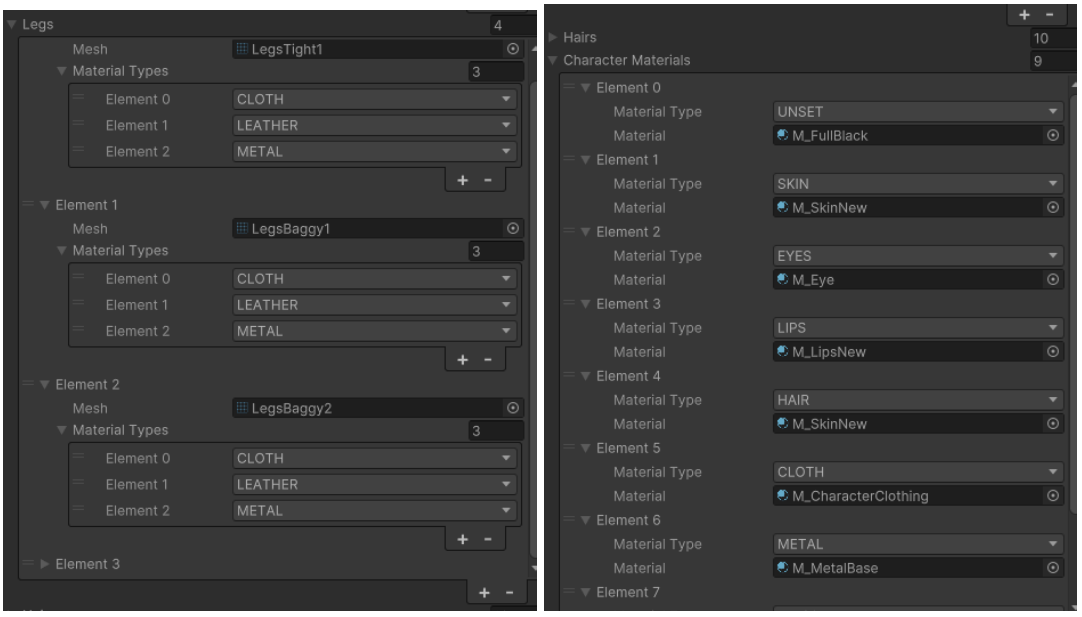
## Character Creation


Character scripts

A character creation system handles the high number of characters required in-game. Characters generate at runtime from a pool of body parts and colours, creating a collection of skinned meshes sharing one Animator.

```
//container for the data used to generate a character
//contains all body part data, the character's stature, and all colors used to color body part materials
[System.Serializable]
public class Body
{
    //scale of the character
    public float stature = 1.0f;
    //all body parts
    public BodyPart head, torso, armL, armR, handL, handR, legs, hair;
    //consistent color for various materials
    public Color skinTone, eyeColor, lipColor, hairColor, metalColor, emissiveColor, leatherColor;
    //various colors for various clothing items
    //arms use same color as torso so they appear as sleeves
    public Color clothColorHead, clothColorTorso, clothColorLegs;
}
```

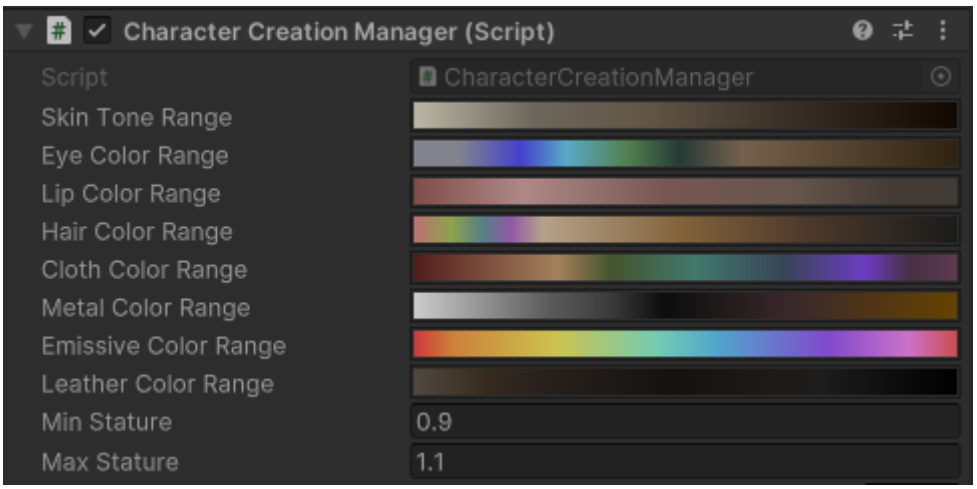
```
//randomly generate a Body for the CharacterCreator to use
public Body GenerateCharacter(Body pregeneratedInfo)
{
    Body newBody;

    //if we already passed in info, leave it alone
    if (pregeneratedInfo != null)
    {
        newBody = pregeneratedInfo;
    }

    //otherwise start from scratch
    else
    {
        newBody = new Body();
    }

    newBody.stature = UnityEngine.Random.Range(minStature, maxStature); //random stature in range
    newBody = GenerateColors(newBody); //generate the color scheme for this character
    newBody = GenerateBodyParts(newBody); //generate all body parts

    return newBody;
}
```




```
//generates and adds one body part skinned mesh to the character
GameObject AddBodyPart(BodyPart bodyPart, string partName)
{
    //create the object and parent it to the character
    GameObject newBodyPartObj = new GameObject(partName);
    newBodyPartObj.transform.SetParent(transform, false);

    //add the skinned mesh
    SkinnedMeshRenderer renderer = newBodyPartObj.AddComponent<SkinnedMeshRenderer>();
    renderer.sharedMesh = bodyPart.mesh;

    //set the bounds as tight as i safely can without knowing the exact proportions of the character
    //these should still be relative to the root bone as all body parts use the same skeleton with the root centered between the character's feet
    renderer.localBounds = new Bounds(new Vector3(0f, 0f, 0f), new Vector3(0.1f, 0.3f, 0.1f));
    //follow the original renderer's rig
    renderer.bones = originalRenderer.bones;
    renderer.rootBone = rootBone;

    //set up the materials needed for this body part
    Material[] materials = new Material[bodyPart.materialTypes.Length];

    for (int i = 0; i < materials.Length; i++)
    {
        materials[i] = CharacterCreationManager.instance.GetBaseMaterialByType(bodyPart.materialTypes[i]);
    }

    //assign the materials
    //would just assign them one by one but renderers don't like that for some reason
    renderer.materials = materials;

    //assign colors to each material
    //using for instead of foreach so i can access both arrays
    for (int i = 0; i < renderer.materials.Length; i++)
    {
        switch (bodyPart.materialTypes[i])
        {
            case EMaterialType.EYES:    //eyes use a custom shader to change just the iris color
                renderer.materials[i].SetColor("_Eye_Color", GetColorByPartMaterial(partName, bodyPart.materialTypes[i]));
                break;
            case EMaterialType.METAL:   //these material types use my master shader graph, so set the Tint property
            case EMaterialType.EMISSIVE:
            case EMaterialType.CLOTH:
                renderer.materials[i].SetColor("_Tint", GetColorByPartMaterial(partName, bodyPart.materialTypes[i]));
                break;
            default:    //by default, assume it's a default unity material and just set the color property
                renderer.materials[i].color = GetColorByPartMaterial(partName, bodyPart.materialTypes[i]);
                break;
        }
    }

    return newBodyPartObj;
}
```

CharacterCreationManager, CharacterCreator

CharacterCreationManager holds part and colour references/data for CharacterCreator. CharacterCreator asks CharacterCreationManager for random parts/colours and generates corresponding skinned meshes.




Gradients

CharacterCreationManager samples colours from intuitive gradients.

## Character Animation

ECharacterState controls character activity animation for easy Inspector modification. When set to a seated activity, characters locate and snap to a nearby seat when spawned, streamlining in-editor character placement.


Blinking

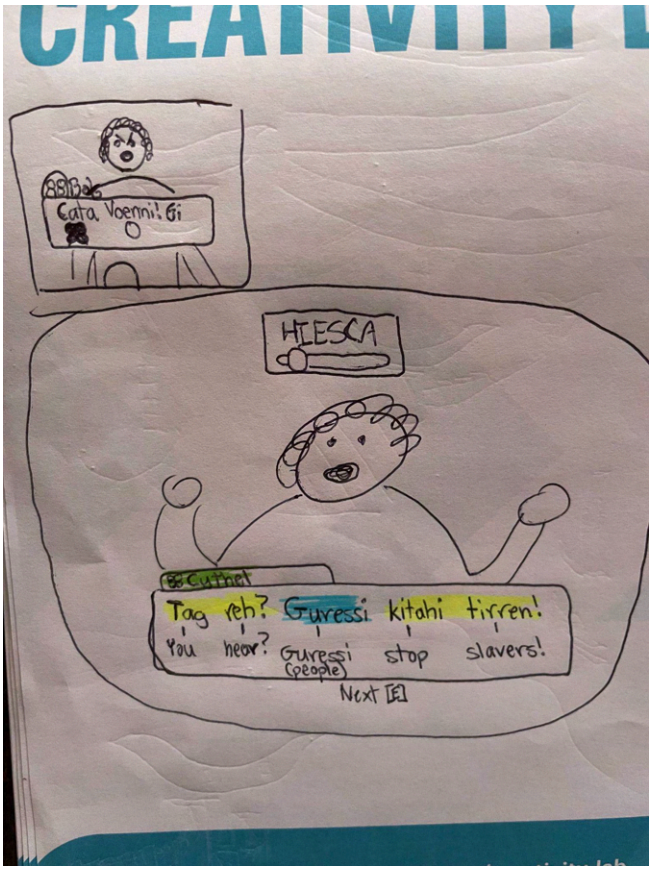
Character faces are rigged for mouth movements and blinking. Characters play a talking animation when conversing with the player and gesture with their arms if not preoccupied with another activity. A blinking animation is layered on top of all other

animations to ensure blinking is not inconsistent depending on the current activity or dialogue animation.


Preoccupied character

Characters' heads use IK to look at the player. Most characters turn to the player when looked at, but others are focused on their activity and only look at the player when conversing. This uses Unity's Animation Rigging package and may be a performance hit.

# Dialogue and Translation





```
//contains data for a single translatable phrase in a single language
[System.Serializable]
public class Phrase
{
    public ELanguage overrideLanguage;  //if not UNSET, this will override the speaker's native language with the desired language

    [TextArea(2,5)]
    public string phrase;   //the text in non-translated language

    [TextArea(2, 5)]
    public string translation;  //the translation of the phrase text

    public string context;  //additional optional context. used to clarify confusing translations (e.g. sayings that don't translate word-for-word to english)
}
```

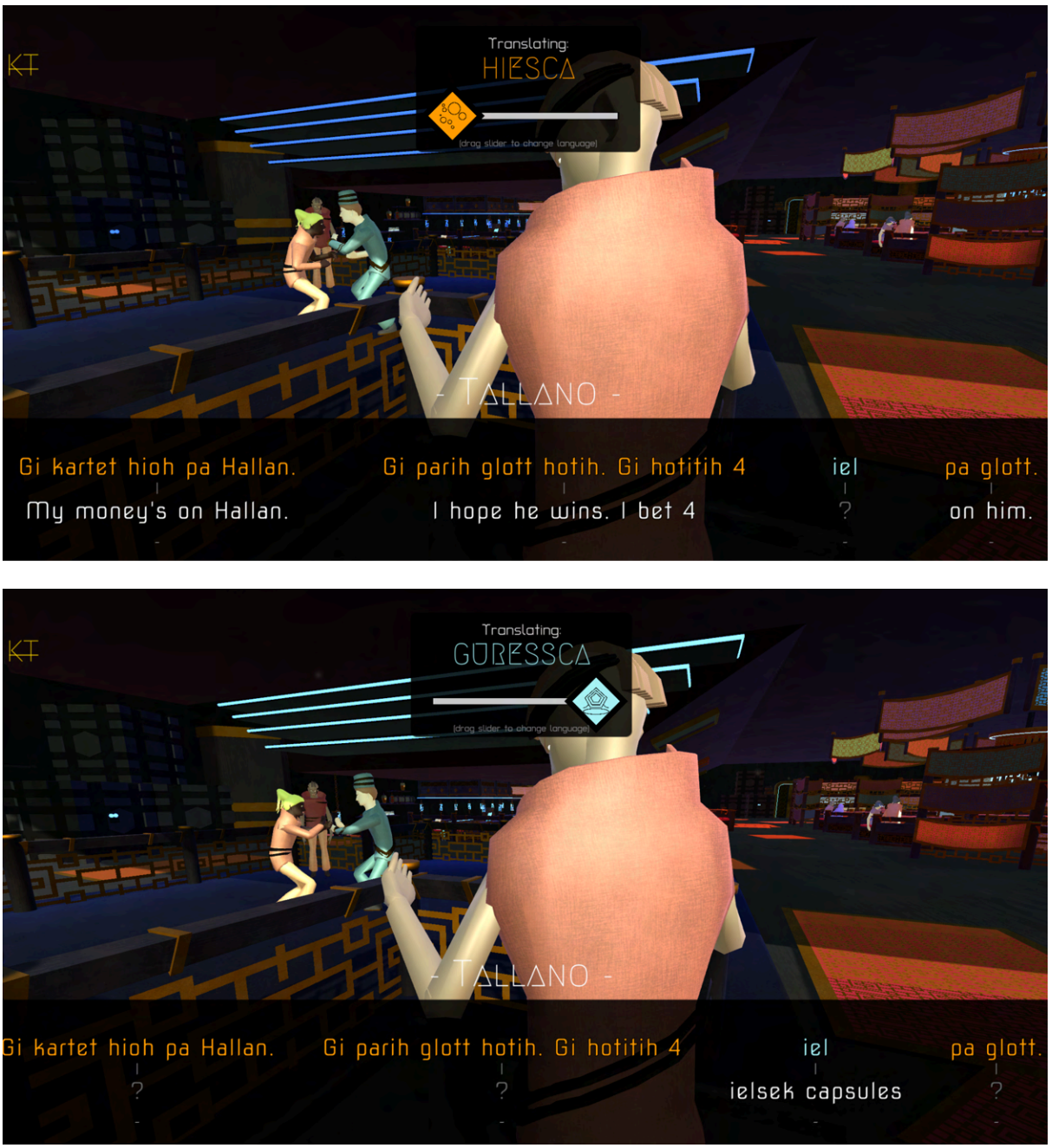
```
//contains data for a single dialogue (one dialogue box on-screen)
[System.Serializable]
public class Dialogue
{
    public ESpeaker speaker;    //where the dialogue is originating from
    public Phrase[] phrases = new Phrase[1];    //the phrases that constitute this dialogue
}
```

Dialogue system

A robust dialogue system handles translating individual phrases across multiple conversations with characters and written text. This structure breaks dialogue into translatable Phrases, single-screen Dialogues, and overarching Conversations. Multiple Conversations per DialogueCharacter allows for narrative progression and repeated final quips, which indicate the character has no more dialogue.


Dialogue auto-progresses in cutscenes

DialogueManager manages the real-time flow and UI of dialogue with public methods like NextDialogue(). These enable compatibility with Signal Emitters and other scripts, meaning conversations can auto-progress in cutscenes where events must be timed.




Translator tuning

The translator uses a slider to simulate “tuning” to a language. Clicking phrases also translates them. When translated, phrases reveal their English translation and additional context if necessary. No translation gameplay is present when the player thinks to themself, representing Quuvol’s inner monologue.

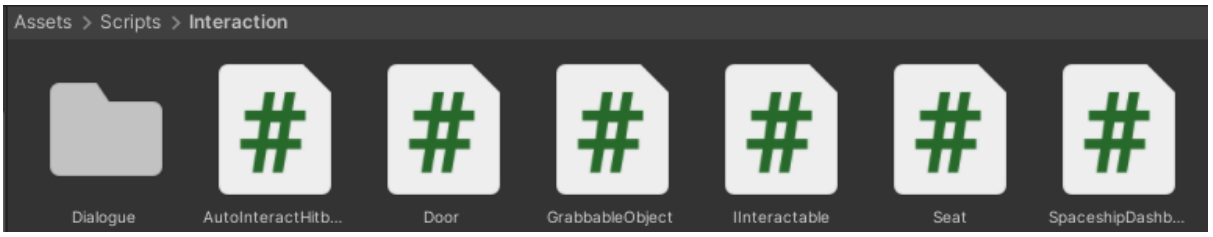
## Player

```
//player movement/action state machine
//switch player state between stuff like normal movement, climbing ladder, sitting down, in dialogue...
//state should NOT be set outside of this function. this is the correct way to update state
public void SetPlayerState(EPlayerState newState)
{
    //PREVIOUS STATE
    switch (state)
    {
        case EPlayerState.SEATED:
            //do NOT leave the seat if talking while seated
            if (newState != EPlayerState.DIALOGUE)
            {
                transform.position = preSeatedPosition;
            }
            break;
        //after cutscene/paused, make inventory visible again and enable camera
        case EPlayerState.CUTSCENE:
        case EPlayerState.PAUSED:
            Inventory.instance.SetVisibility(true);
            if (cam != null)
            {
                cam.gameObject.SetActive(true);
            }
            break;
        default:
            break;
    }

    state = newState;

    //NEW STATE
    switch (state)
    {
        //before cutscene, hide inventory and disable camera
        case EPlayerState.CUTSCENE:
            Inventory.instance.SetVisibility(false);
            camController.SetMouseVisibility(false, false);
            if (cam != null)
            {
                cam.gameObject.SetActive(false);
            }
            break;
        case EPlayerState.PAUSED:
            Inventory.instance.SetVisibility(false);
            camController.SetMouseVisibility(true, false);
            break;
        case EPlayerState.DIALOGUE:
            camController.SetMouseVisibility(true, false);
            break;
        case EPlayerState.MOVABLE:
        case EPlayerState.LADDER:
            camController.SetMouseVisibility(false, true);
            break;
        //quickly save the player's position before snapping to seat
        case EPlayerState.SEATED:
            preSeatedPosition = transform.position;
            camController.SetMouseVisibility(false, true);
            break;
    }
}
```

PlayerMovement

PlayerMovement and CameraController handle the player/camera using a state machine. This includes first-person movement, climbing ladders, and immobile states like cutscenes.

## Interaction




```
//types of interactions
//UI script interprets these and uses them to prompt the player
//e.g. DIALOGUE = "Talk [E]", READABLE = "Read [E]" on the UI
public enum EInteractionType
{
    GENERIC,    //Interact
    DIALOGUE,   //Talk
    READABLE,   //Read
    GRABBABLE,  //Grab/Pick up
    SEAT,       //Sit
    DOOR,       //Door
    SPACESHIPDEPART, //Leave Kallay Tirridor from the Kit Orel dashboard
    INSPECTABLE //Inspect something
}
```

IInteractable

The IInteractable interface facilitates most interactions: entering conversation, sitting, grabbing items, opening doors, inspecting/reading, etc. The camera raycasts each frame, attempting to target an interactable. When successful, a UI prompt appears with the object's name, the performable action, and any cost the action will incur. IInteractable.Interact() runs on the player's input.


Character faces player

IInteractables receive an OnTargeted() event, allowing characters to face players when targeted.


Purchasing a drink

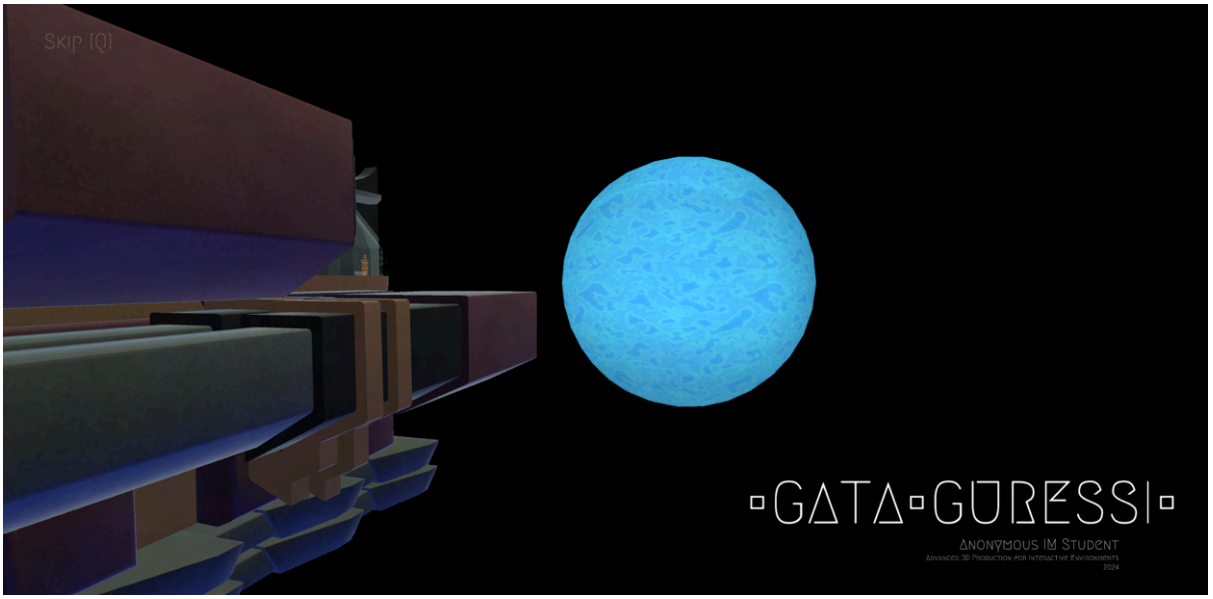
GrabbableObjects snap to the player's "hand" when interacted with and simulate physics when dropped. GrabbableObjects can also trigger dialogue when first grabbed.

Drinks at the conveyor bar move along a spline. Seats attach the player to the seat, disabling movement but allowing free camera rotation.


Interrupting Bila Gola

AutoInteractHitbox triggers an interaction when entered. Walking onto the band's stage repeatedly triggers an argument with the harpist.

## Cutscenes




End credits

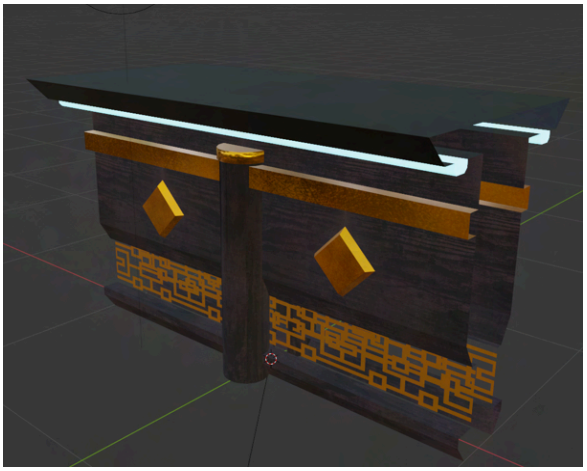
There are two cutscenes made using Unity's Timeline. Signal Emitters call events on fading UI, dialogue, and the player's spaceship for dynamic effects. The space scene plays inside an inverted sphere with scaling spheres for planets.

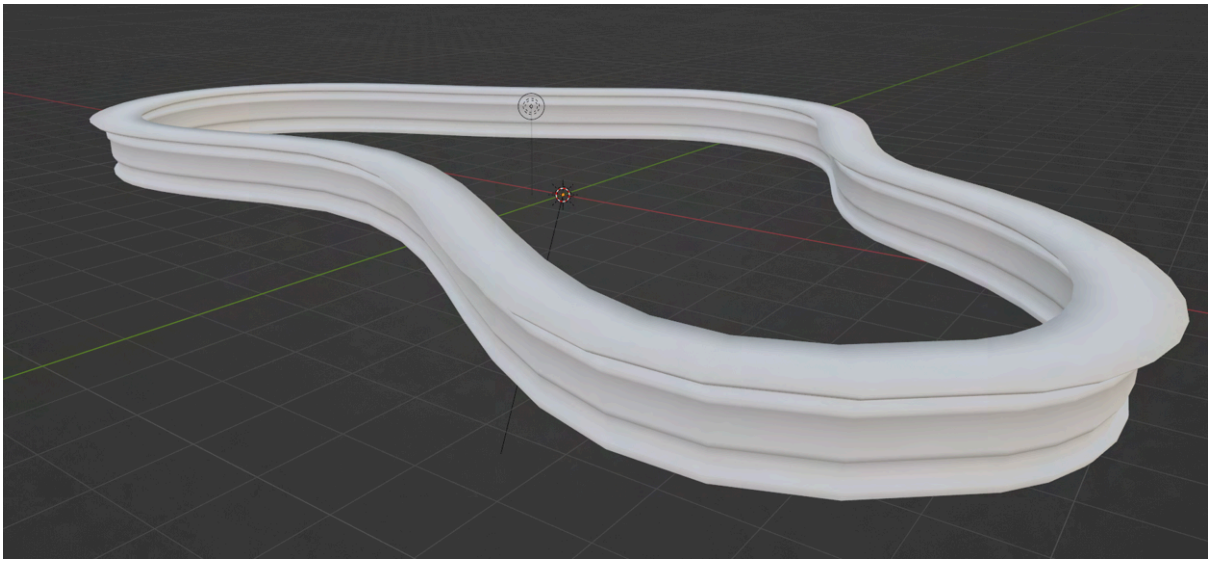
## Optimization

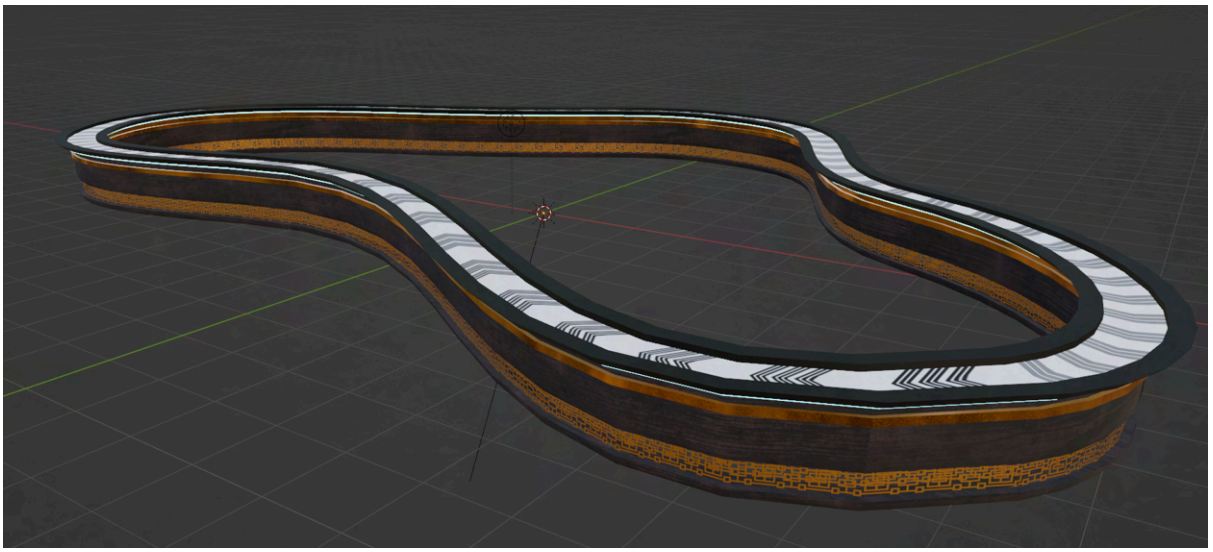
Optimization was the greatest challenge of this project. With the target device in mind, I had to spend longer optimizing than modelling.

Low-poly models

Kallay's geometric aesthetic complemented low-poly modelling. Models are as low-poly as reasonable. Bevelled edges prevent sharp, unnatural lines on objects the player may examine.







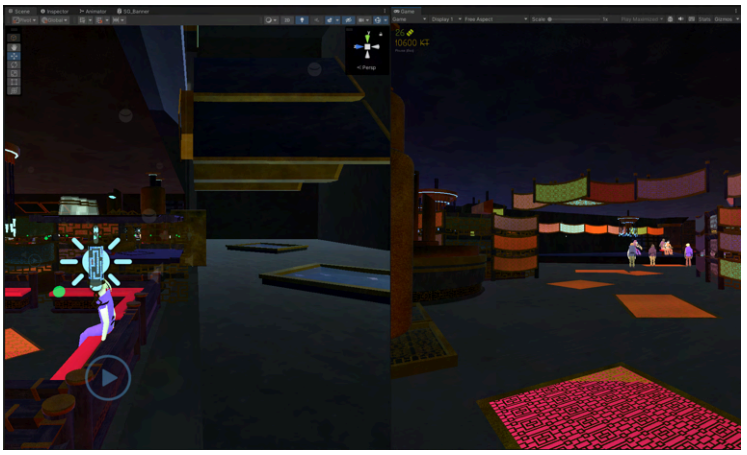
Profile, curve

The upstairs bar was modelled using a bar profile along a curve and animated with a panning material.


Glowing characters

Baked lighting with reflection and light probes light the scene and skinned meshes, though full parity between baked and light probe lighting is impossible. Meshes with vertex position shaders (e.g. banners) cannot be fully static but still use baked lighting.

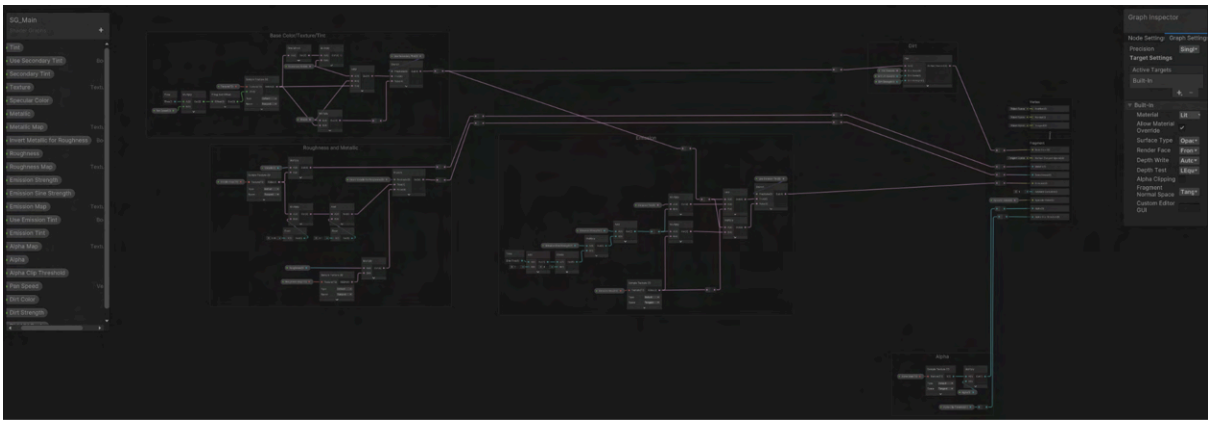




Area loading (scene view left, game view right)

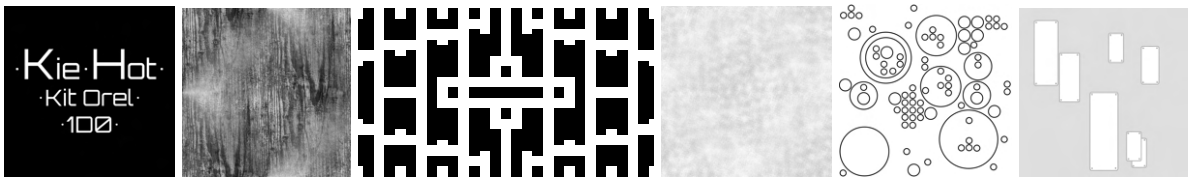
AreaLoadManager activates characters and details when approximately visible to the player. Opening a door or entering an area loads the area and adjacent areas and unloads unnecessary areas. Occlusion culling only affects static objects, so AreaLoadManager was necessary for culling characters and rigidbodies.

## Shaders, Materials, and Textures



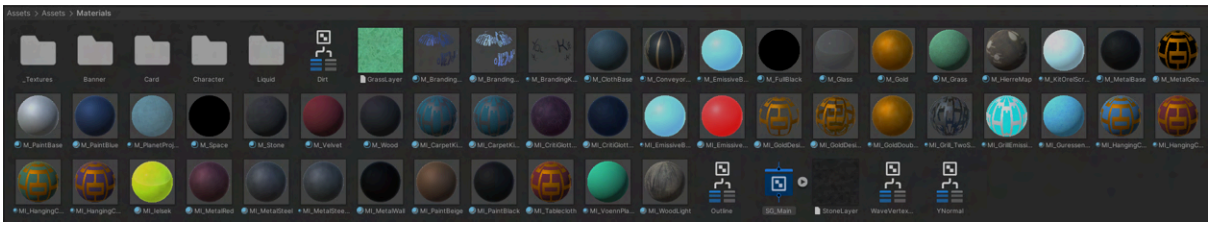
SG_Main

Using Unity's shader graph, I implemented SG_Main, a master shader that facilitates material creation and saves resources.



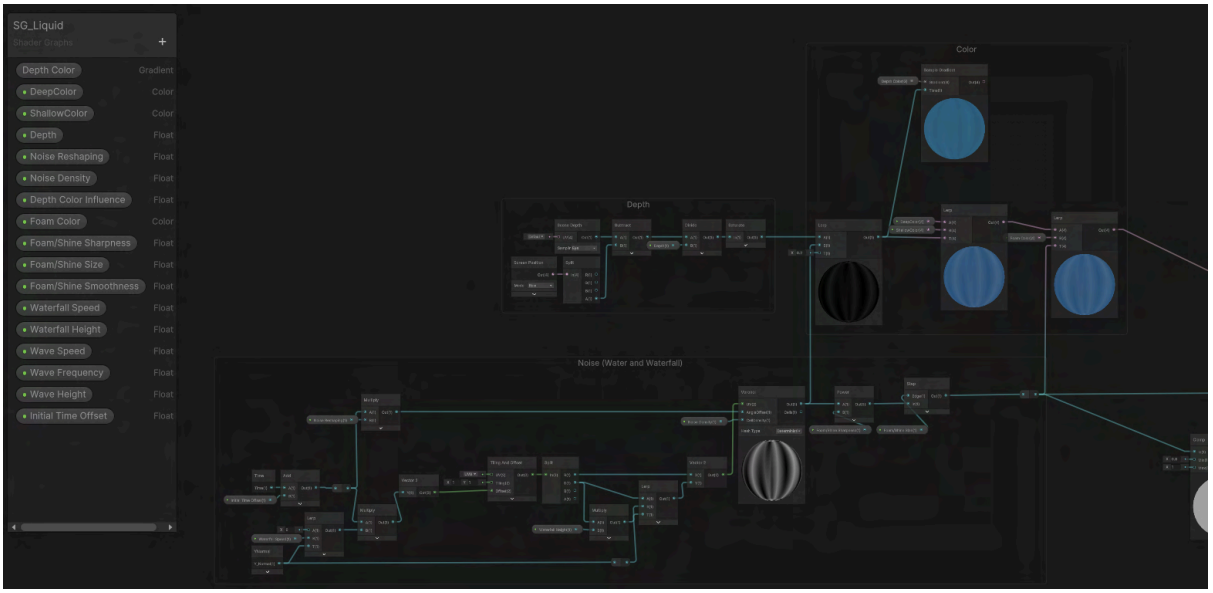

Greyscale textures

Greyscale textures are tinted or used as alpha maps across multiple materials. This enables tint/metallic/roughness variation and "layering" an alpha-clipped texture over a base colour. SG_Main supports pulsing emission, panning, and tiling configurable dirt for realism.



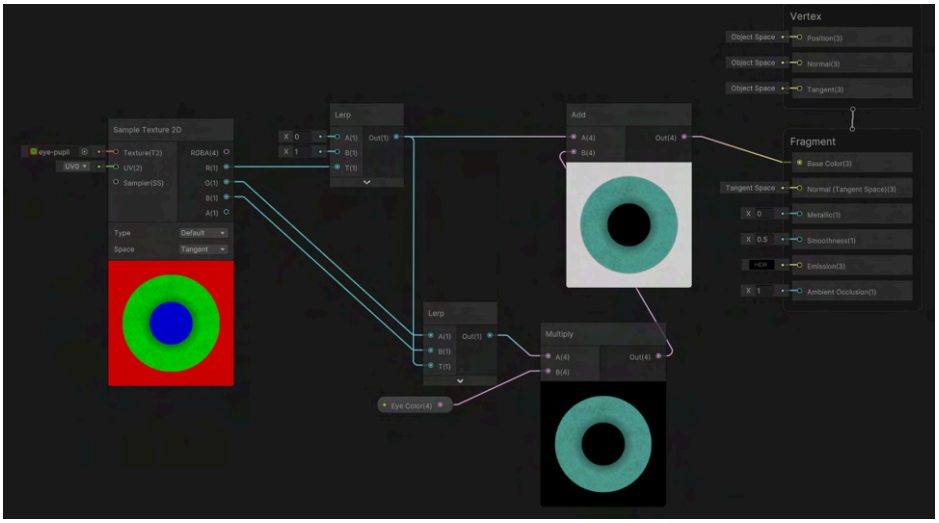
Materials/shaders

Material variants were used in a hierarchy where changes to base materials affect all children. SG_Main replaces Unity's default Lit shader for most materials.



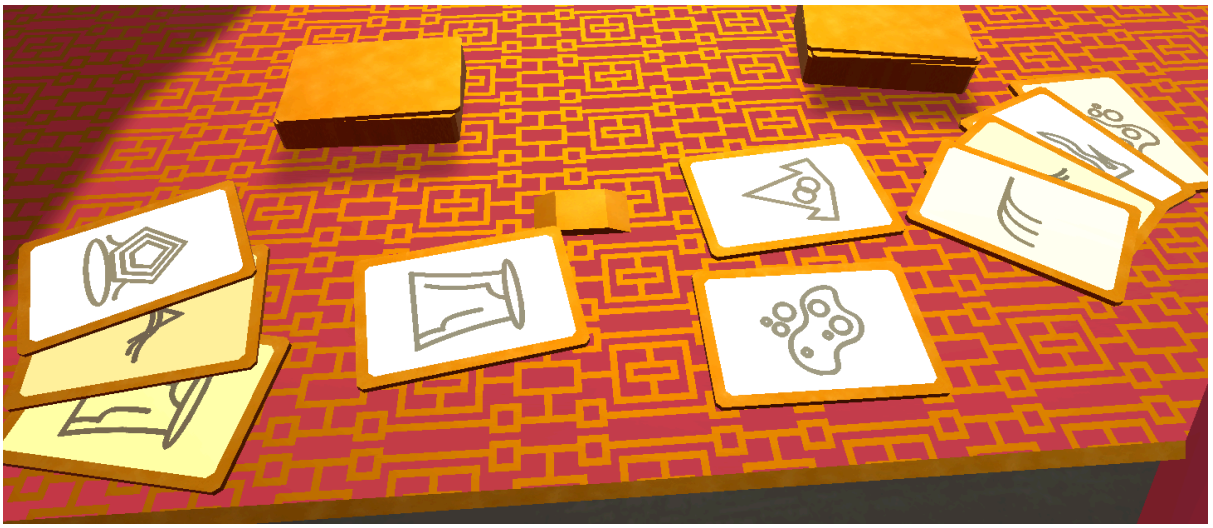

SG_Liquid

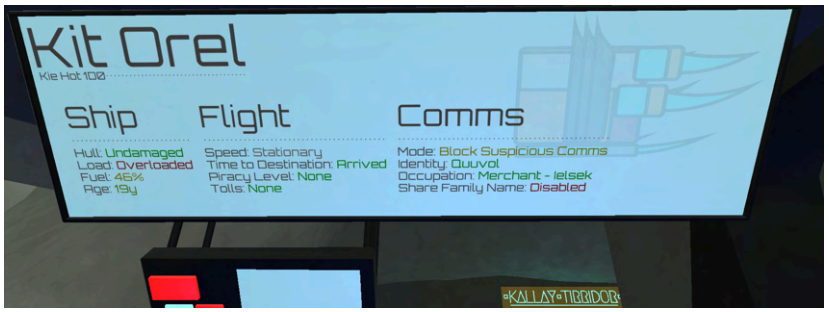
Liquid and flowing banners use separate shaders to prevent a performance hit to SG_Main. SG_Liquid changes colour by screen depth, creates water foam, and stretches along the Y-axis for waterfalls. SG_Liquid and SG_Banner use WaveVertexOffset, a subgraph that offsets vertices for a flowing wave effect.

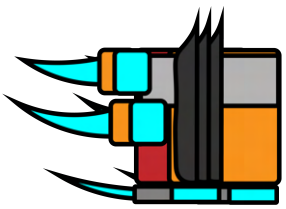



SG_Eye

SG_Eye uses an RGB eye texture to give characters different eye colours.

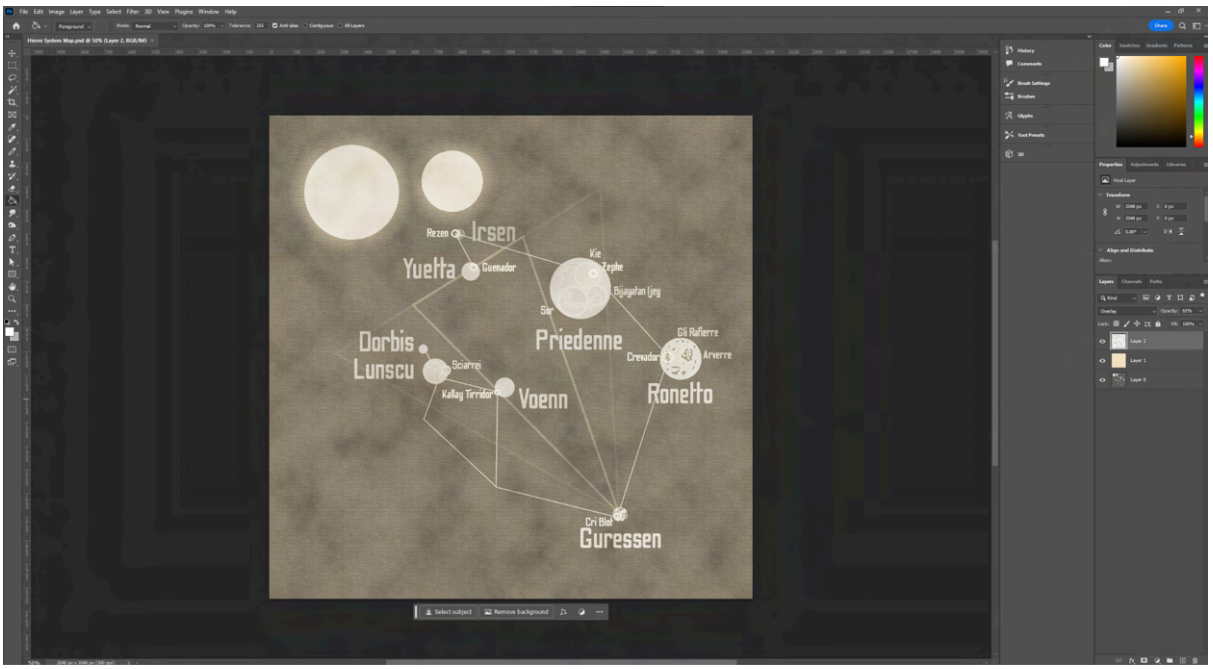





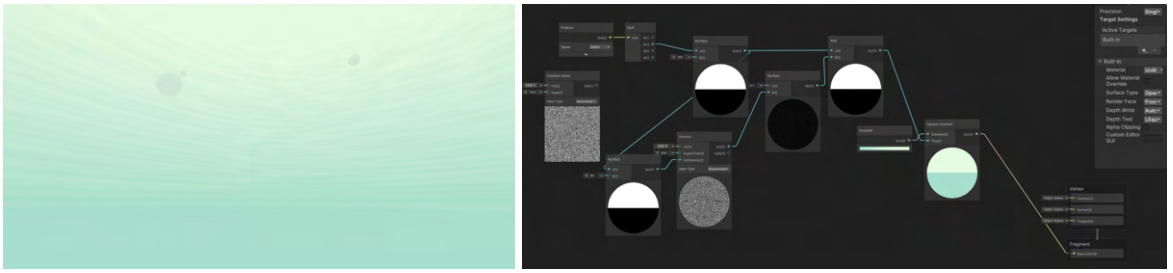
Icons

Each region/language has an icon used on banners, playing cards, and the translator. The player's spaceship dashboard displays an icon representing the ship.




Planetary map

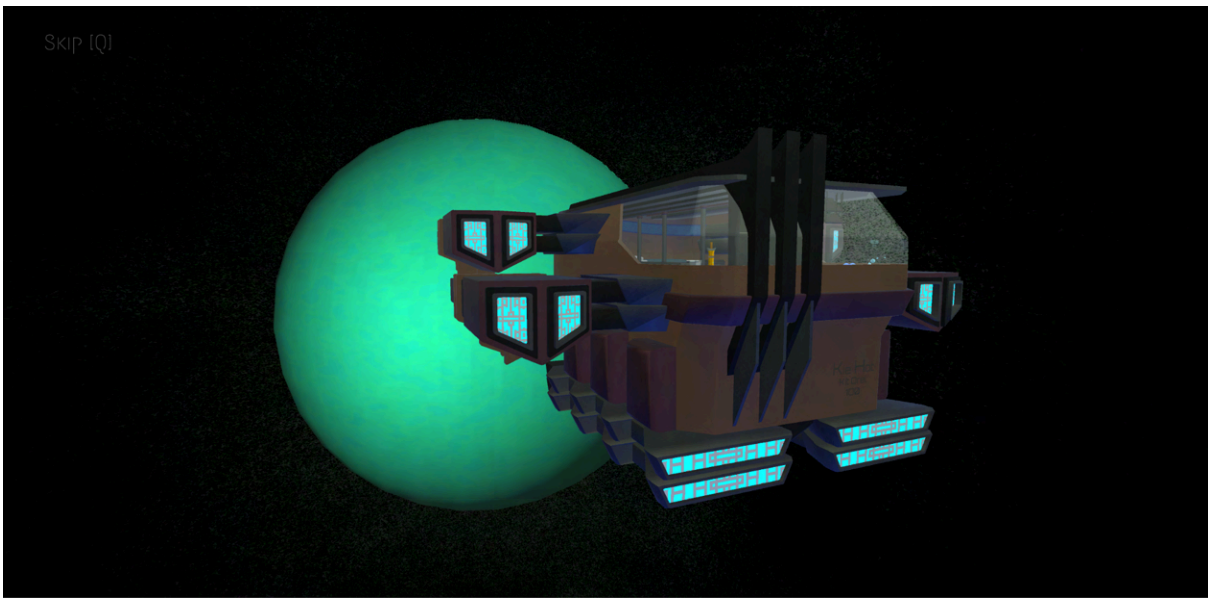
The map is a desaturated Hierre map beneath Photoshop's Clouds and Texturizer filters.



Skybox

In a separate scene, I wrote a cloud/sky shader and applied it to an inverted sphere. Spheres representing nearby planets use low-opacity materials, so clouds appear closer. Using a reflection probe, I rendered the scene to a cubemap for the game's skybox.

# Reflection




Leaving Voenn

*Gata Guressi* successfully met its goals. Kallay Tirridor mixes a sci-fi setting with an inviting bar aesthetic, just as intended. Its spaceships, sculptures, and geometric textures create a unique atmosphere. I achieved the desired level of cohesion and immersion through worldbuilding and dialogue. The cutscenes boost the game's quality and circumvent the game feeling like a one-off experience. I could improve real-time immersion by giving characters more animations: friends conversing, waiters serving drinks, and bartenders preparing drinks and polishing glasses.


No "drink" option???

My most unforgivable shortcoming was omitting the option to drink a held beverage.


Not everyone knows Hiesca…

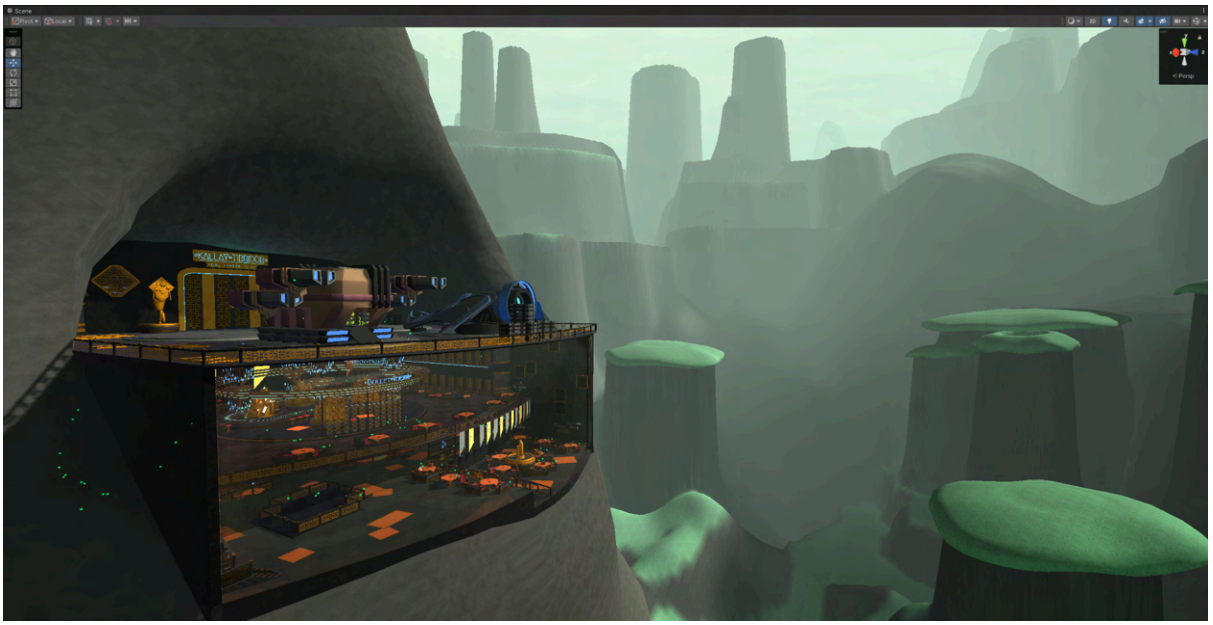
The translation system builds immersion and grants insight into characters' culture. I was amused to find myself nearly fluent in Hiesca due to the amount of dialogue I wrote. In the future, I would expand on this system by making greater use of the planetary languages, as many conversations are entirely in Hiesca, and adding observable conversations between other characters. Nonlinear dialogue that varies based on the player's held item or financial status would boost immersion.

Empty tables

SFX are needed for footsteps, ielsek sales, spaceship blastoff, and characters. Many characters still need dialogue, and the environment lacks detail. Character clothing options are very limited. Other additions are necessary for polish: unique guard clothing, interaction raycast bugfixes, and light probe improvements.

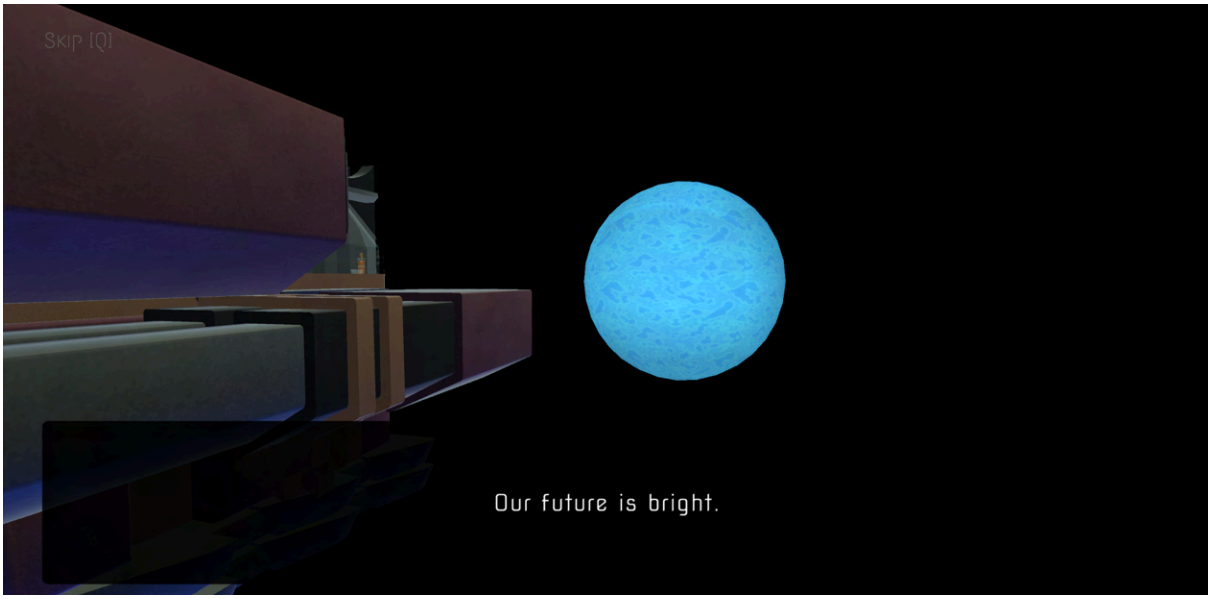

Mesh clipping

CharacterCreationManager's material type system could use some improvements, but the system is prime for additions: more parts, culture- and class-specific clothing, and a weighted body part system where some parts appear more frequently than others. Android characters are a major facet of the universe but were omitted from the game. Many body parts have weight paint issues (e.g. characters blink and their hair scrunches down).



Kallay Tirridor in-editor

Though my optimisation efforts were essential for the game's playability, the framerate on the lab machines was disappointing at best. Regardless, I am pleased with the resulting performance gains. I would further optimise character creation, but am unaware of any other solutions after extensive research. I could optimise meshes by baking their materials in Blender and slimming down SG_Main. I could also change my target platform; an HDRP build would breathe life into the flat lighting.




Quuvol's monologue

I am very pleased with *Gata Guressi* and believe it warrants future development. Its dialogue and worldbuilding surpassed my expectations, creating the cohesive, immersive, and optimized experience I desired.

Word count: 2737

# References


Arache. (2023). *I will make spaceship concept art and spaceship for you* [Image]. Available at: https://www.fiverr.com/aracheweoca/make-spaceship-concept-art-and-spaceship-for-you [Accessed 16 January 2024].

Arkane. (2017). *Prey* [Video game].

Capcom. (2018). *Monster Hunter World* [Video game].

Capcom. (2020). *Monster Hunter World: Iceborne*. [Expansion pack].

CD Projekt Red. (2020). *Cyberpunk 2077*. [Video game].

chungking. *Modern airport scene, view of the terminal window — Photo* [Image]. Available at: https://depositphotos.com/photo/modern-airport-scene-view-terminal-window-228169494.html [Accessed 16 January 2024].

Disasterpeace. (2016). *Hyper Light Drifter*. [Soundtrack]. Disasterpeace.

Fexelea. (2021). *Namielle Alpha + Armor Set.* [Image]. Available at: https://monsterhunterworld.wiki.fextralife.com/Namielle+Alpha+++Armor+Set [Accessed 16 January 2024].

Fieschi, P. (2021). *Boxy Stripy Ship 2.* [Image]. Available at: https://pierreef.artstation.com/projects/3d99XD [Accessed 16 January 2024].

Hadad, A. et al. (2023). *Japantown - Westbrook Apartment* [Image]. Available at: https://www.ign.com/wikis/cyberpunk-2077/Japantown_-_Westbrook_Apartment [Accessed 16 January 2024].

HBO. (2011). *Game of Thrones* [Television series].

Heart Machine. (2016). *Hyper Light Drifter.* [Video game].

Hello Games. (2016). *No Man's Sky*. [Video game].

Jorge, I. (2023). *Eataly London* [Image]. Available at: https://www.google.com/maps/place/Eataly+London/@51.5187586,-0.0801718,3a,75y,90t/data=!3m8!1e2!3m6!1sAF1QipOncP4xbfmSHPKpGXwlXGznFKqggAzis2V5CGoL!2e10!3e12!6shttps:%2F%2Flh5.googleusercontent.com%2Fp%2FAF1QipOncP4xbfmSHPKpGXwlXGznFKqggAzis2V5CGoL%3Dw203-h270-k-no!7i3024!8i4032!4m18!1m8!3m7!1s0x4876034c44354bb1:0xc57f7d6c0b61ff5f!2sLiverpool+Street!8m2!3d51.5187516!4d-0.0814374!10e5!16zL20vMHAwanQ!3m8!1s0x48761d5c5da428a7:0x1e1992afbd13c552!8m2!3d51.5187586!4d-0.0801718!10e5!14m1!1BCgIgAQ!16s%2Fg%2F11mpsdd7vk?entry=ttu [Accessed 16 January 2024].

Knakveey. *02 02 - Prey ConceptLobby GC16* [Image]. Available at: https://prey.fandom.com/wiki/Talos_I_Lobby?file=02_02_-_Prey_ConceptLobby_GC16.jpg [Accessed 16 January 2024].

kotakuinternational. (2017). *If You're Playing Prey, Try Turning Off Mission Objectives* [Image]. Available at:

https://www.kotaku.com.au/2017/05/if-youre-playing-prey-try-turning-off-mission-objectives/ [Accessed 16 January 2024].

Leete, R. I. (2022). *The Space Age Aesthetic: Influencing Architecture and Interiors* [Image]. Available at: https://www.archdaily.com/981405/the-space-age-aesthetic [Accessed 16 January 2024].

Lucasfilm Ltd. (1977). *Star Wars*. [Franchise].

Moongrease. (2020). *Clean T1 Halo Long nose with box thrusters in White and very pale Blue/Grey. T3 economy Euclid.* [Image]. Available at: https://www.reddit.com/r/NMSCoordinateExchange/comments/j2lpd9/clean_t1_halo_long_nose_with_box_thrusters_in/ [Accessed 16 January 2024].

Oskwarek, Z. (2023). *Oga's Cantina in Star Wars Galaxy's Edge (Menu, Review and Reservations)* [Image]. Available at: https://ziggyknowsdisney.com/ogas-cantina-star-wars-galaxys-edge/ [Accessed 16 January 2024].

pRopaaNS. (2022). *System Frigate / Galleon , Cargo Pods , Wedge , Keel / light Turquoise, Gray / Pirate Vy'Keen system* [Image]. Available at: https://www.reddit.com/r/NMSCoordinateExchange/comments/ypzueg/system_frigate_galleon_cargo_pods_wedge_keel/ [Accessed 16 January 2024].

Rutter, C. and Kent, R. (2007). *Conveyor belt sushi at Bluewater Shopping Centre in England* [Image]. Available at: https://en.wikipedia.org/wiki/Conveyor_belt_sushi#/media/File:Bluewater3753.JPG [Accessed 16 January 2024].

Samayoa, M. (2023). *Portland's first electric garbage truck will hit eastside streets soon* [Image]. Available at: https://www.opb.org/article/2023/11/02/portland-oregon-electric-garbage-truck-vehicle-environment-emissions/ [Accessed 16 January 2024].

Uglow, J. *Millennium Falcon Interior Environment Art* [Image]. Available at: https://juglow95.artstation.com/projects/Z5J9rR [Accessed 16 January 2024].

Valyrian Wildfire. *Titan of Braavos* [Image]. Available at: https://gameofthrones.fandom.com/wiki/Braavos?file=Titan+of+Braavos.jpg [Accessed 16 January 2024].

Zan. (2023). *Арт-директор Cyberpunk 2077 Призрачная свобода: "Фантазия разработчиков следующей части будет ограничена лишь бюджетом и временем"* [Image]. Available at: https://www.goha.ru/art-direktor-cyberpunk-2077-prizrachnaya-svoboda-fantaziya-razrnabotchikov-sleduyushhej-chasti-budet-razrabotchikov-lish-byudzhetom-i-vremenem-xRWEmK [Accessed 16 January 2024].

# Appendices

## Appendix A: Dictionary

**REPORT DICTIONARY**

Here are some fictional terms used in the report that may be confusing/hard to remember:

**Dorbis** - orange bubbling planet. Moon of Lunscu.

**Guressen** - icy planet. Quuvol's home.

**Hal** - blue/black spaceship parked beside the player in the landing dock.

**Hierre** - the star system.

**Hierre Kie** - northern region of planets (Yuetta, Irsen, Priedenne, Ronetto).

**Hierre Sor** - southern region of planets (Voenn, Guressen, Lunscu, Dorbis).

**Hiesca** - the common language.

**Ielsek** - resource the player is selling.

**Irsen** - lava planet.

**Iusol** - current leader of Guressen. Outlawed slavery.

**Kallay Tirridor** (or just "**Kallay**") - trade port/bar the game takes place in.

**Kit Orel** - the player's spaceship.

**Lunsca** - Lunscu's planetary language.

**Lunscu** - pink bubbling planet.

**Priedenne** - savannah planet.

**Quuvol** - the protagonist.

**Ronetto** - desert planet.

**Sorpri** (abbv. of **Sorpriedesca**) - local language of Southern Priedenne.

**Voenn** - grassy/stoney planet. The game takes place here.

**Withol** - former leader of Guressen. Endorsed slavery.

**Yuetta** - water planet.

## Appendix B: Asset References

| Description | Image | Source | License/permission |
|---|---|---|---|
| Stylized Water Shader Graph<br><br>My “WaveVertexOffset” shader subgraph is based on this. | - | BinaryLunar<br><br>https://youtu.be/1yevpCAA_rU?si=MP7_8lkqITUSdQxb | YouTube tutorial |