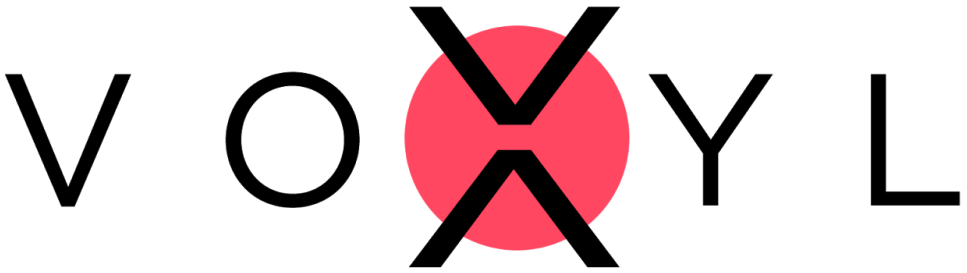


Alex B., Josh C., Kaley K., Hana L.

Share your vision. Get your shot.

1. Problem and Solution Overview

The problem that we identified from our needfinding was one that many visual artists face: communicating their creative visions with the people they need to collaborate with to make their visions come to life. When an artist is in the early stages of planning a project, they often have an idea of what they want their final piece to include, but they may struggle to communicate to their collaborators how all of the parts of their vision are meant to fit together.

With Voxyl, the artist can easily and quickly assemble a 3D moodboard in virtual reality containing all of the images that inspired their vision, as well as lighting and audio to set the mood. Using a convenient 360-degree radial menu at their feet, they can add and delete objects from their scene, switch the terrain between a selection of virtual landscapes, drag and drop objects to reposition them in the scene, and control all aspects of their vision, from the positioning and intensity of light sources to the volume of audio to the size of each of their images.

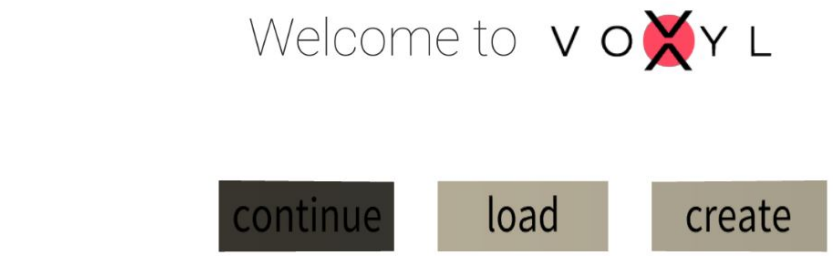
(Fig. 1.1) A scene we built using Voxyl, pictured in Unity (not in virtual reality).

2. Task and Final Interface Scenarios

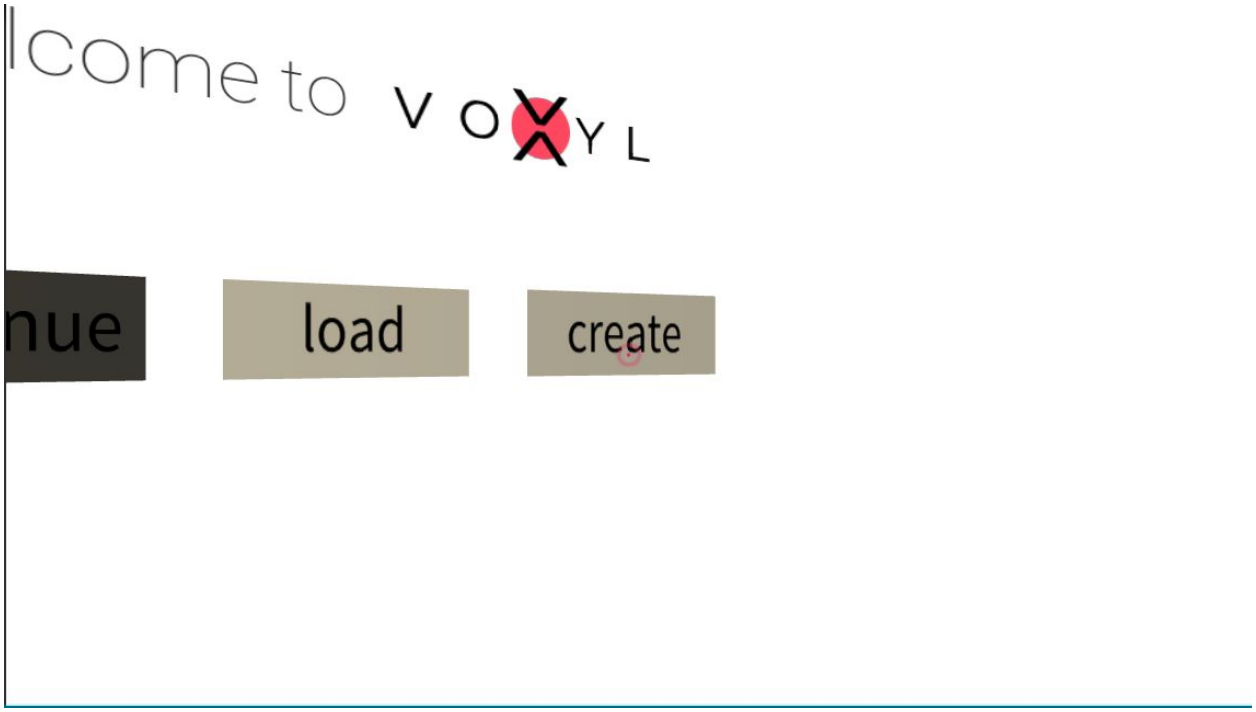
Task 1: Create a new scene

The number one priority in getting artists visions in the hands and minds of others. Being able to create and start a project.

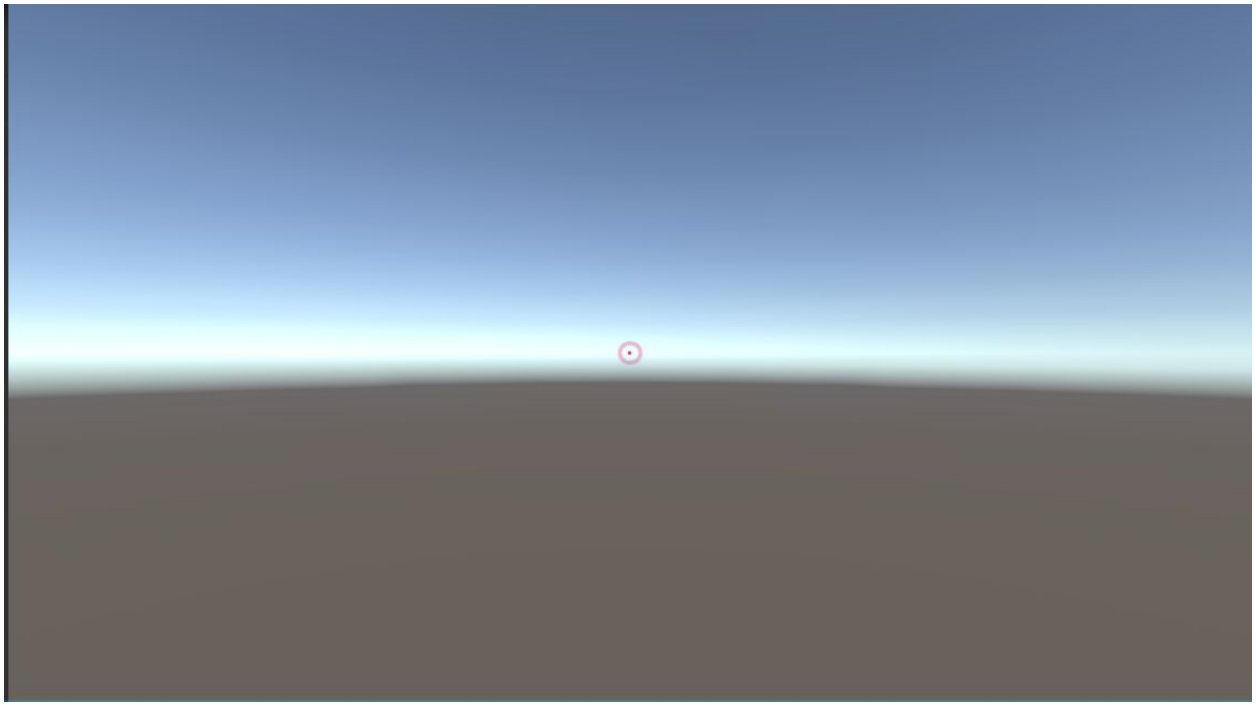
(Fig. 2.1.1) When opening the app you're greeted by this splash screen.



(Fig. 2.1.2) You're then brought into a new scene where you see our home menu.



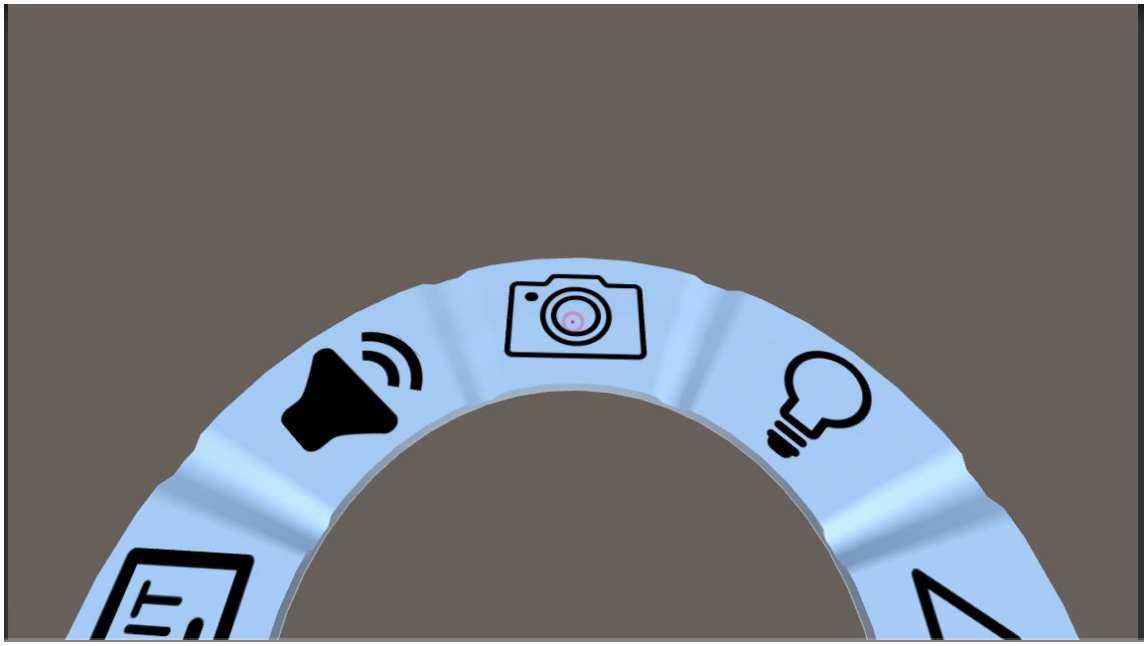
(Fig. 2.1.3) To complete the first task, move your head and the reticle over the "create" button and tap and hold.



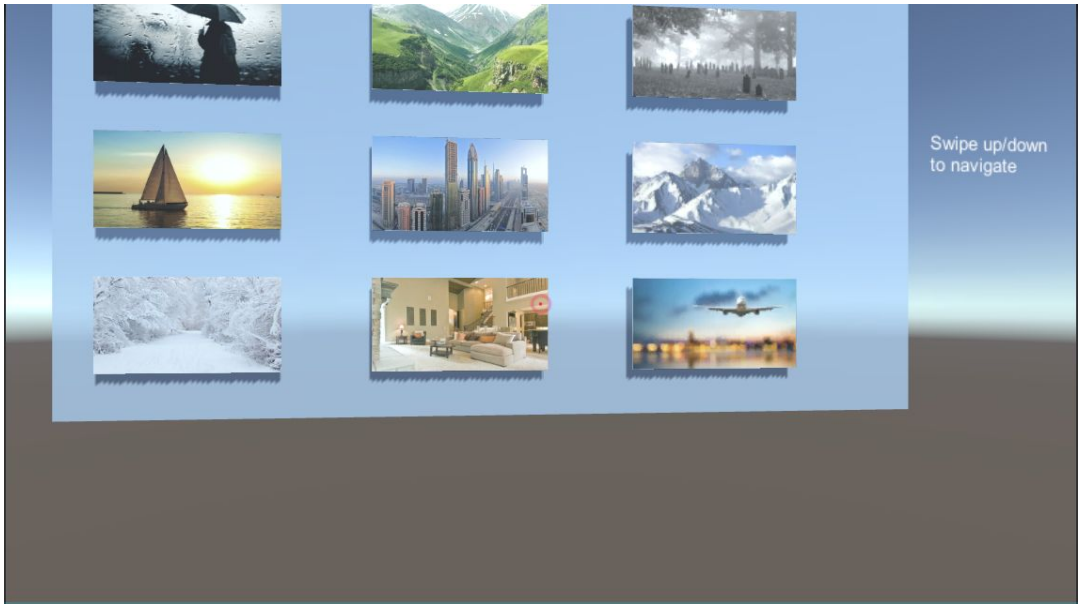
(Fig. 2.1.4) This brings you to a new empty scene with a sky box. Task one complete!

Task 2: Add and reposition media in your scene

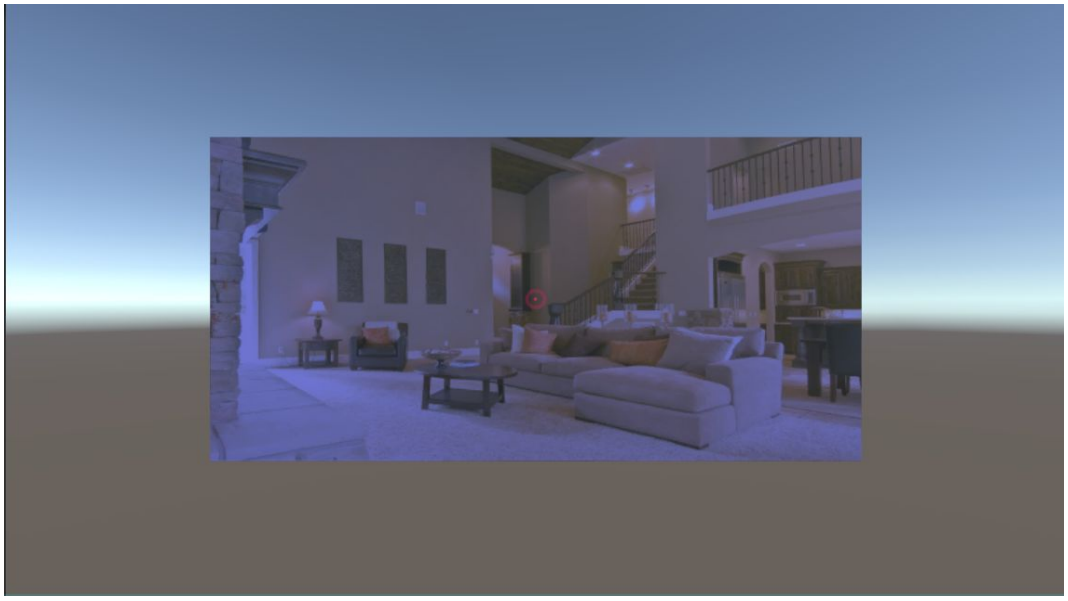
The most basic component of any moodboard is adding photo media and positioning them in a space. This is task 2.



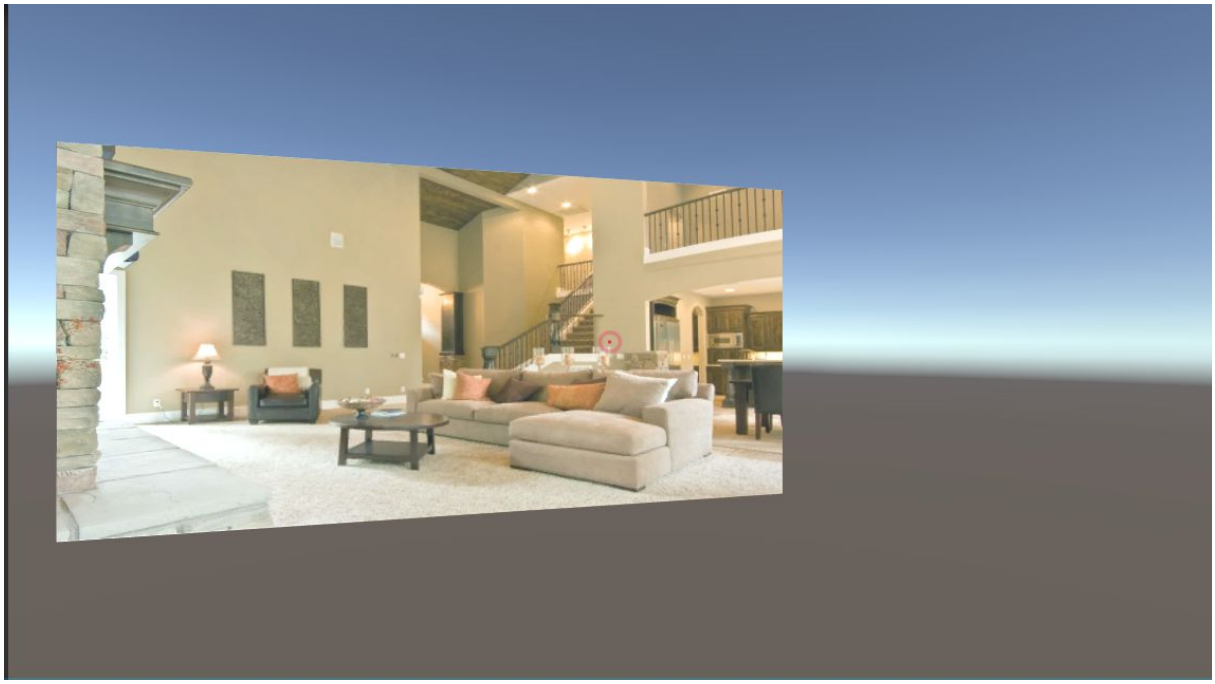
(Fig. 2.2.1) In your new scene, look down at your feet to find the circular menu. Double click the camera icon to pull up the media panel.



(Fig. 2.2.2) Look up from the menu to find a media panel. Scroll through the various imported media using your trackpad functionality, and when you find an image you like, double click to bring it into the scene.



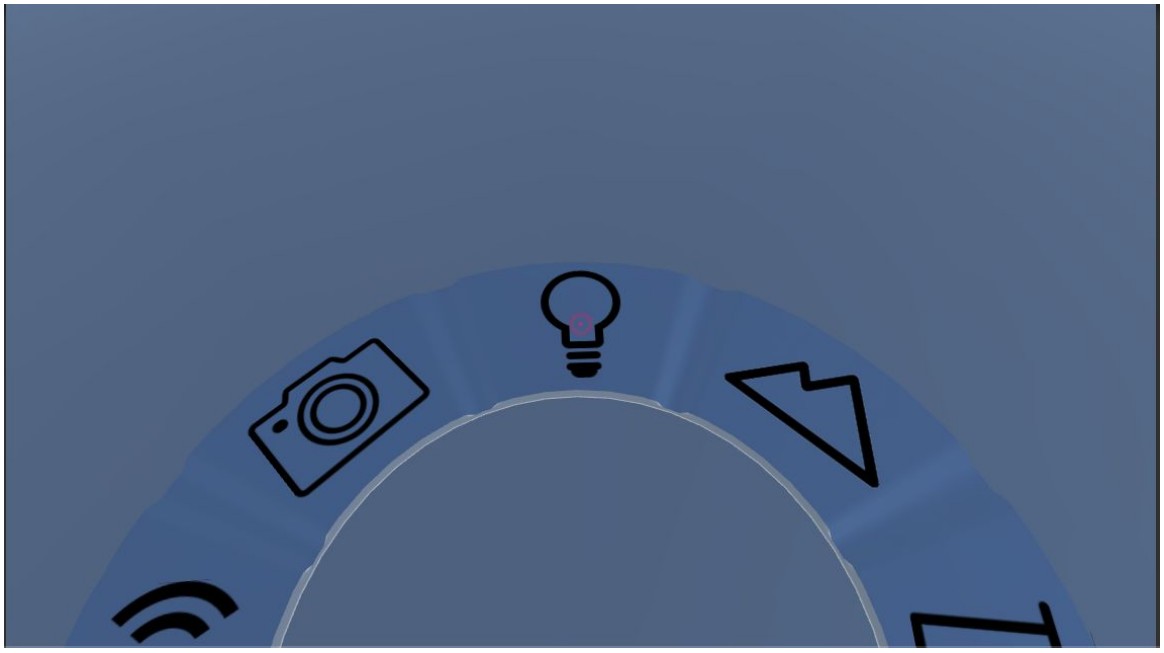
(Fig. 2.2.3) With your image in the scene, double tap it to select and it will tint blue. Move your head to reposition it and scroll on the track pad to change size and move the media forwards and backwards.



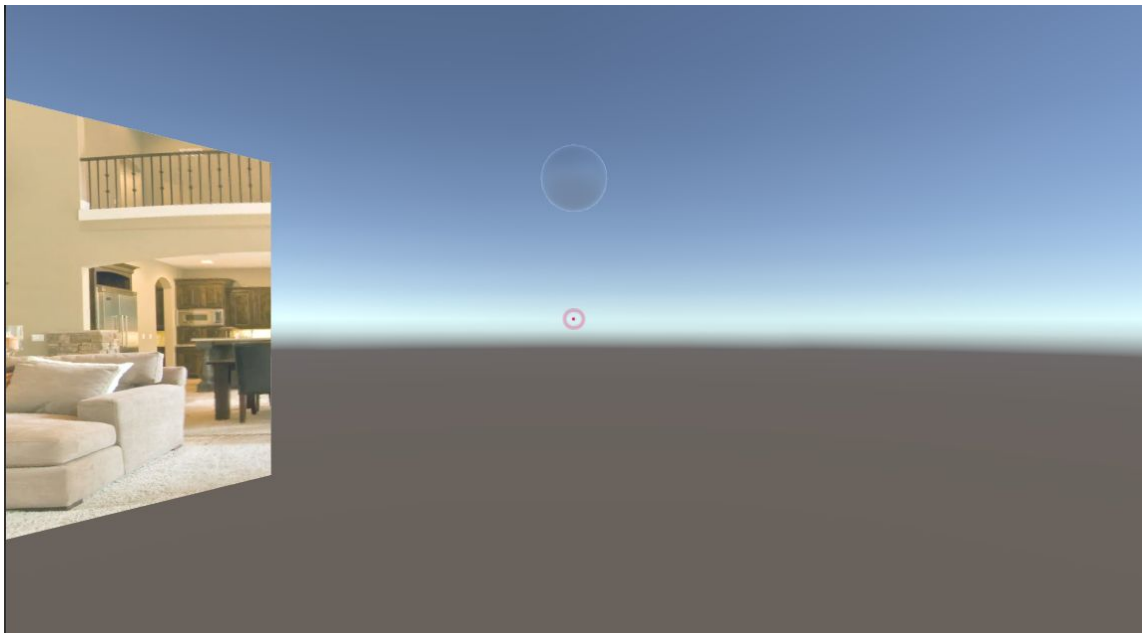
(Fig. 2.2.4) Double tap again to drop your media in the scene. Task 2 complete!

Task 3: Create ambiance through lighting, terrain, and audio. Save, share and exit.

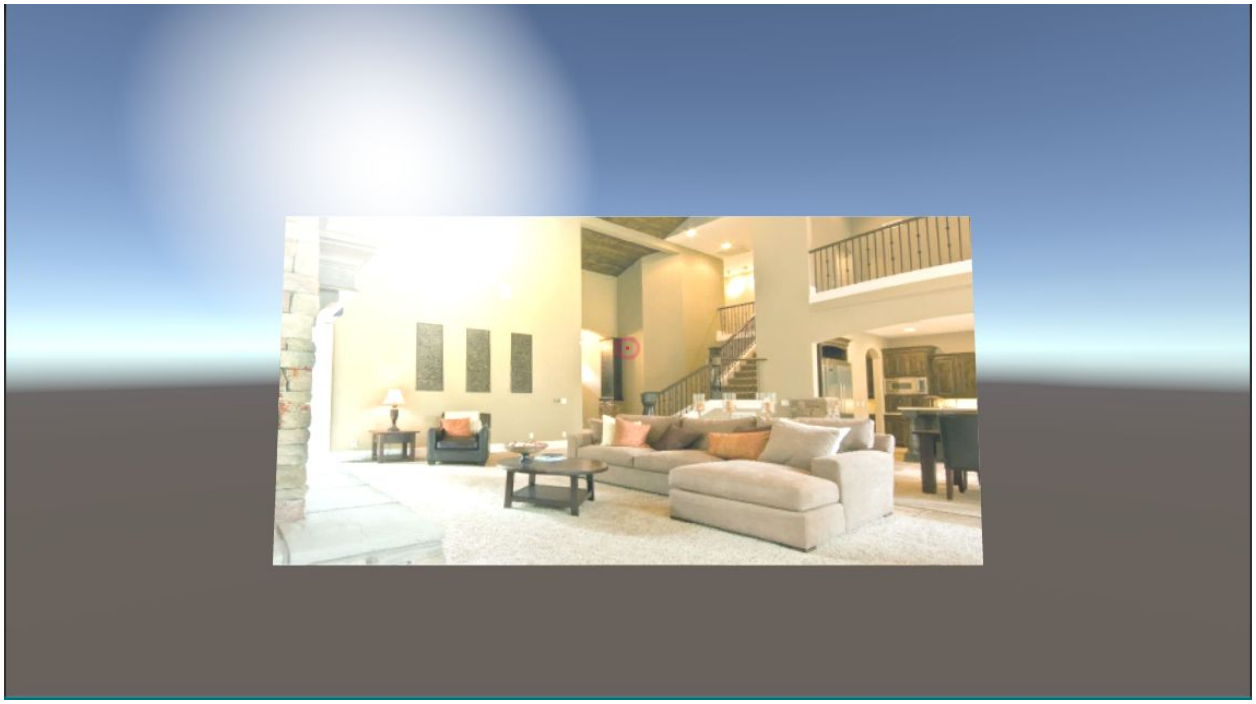
Once the basics are out of the way, there's room to make the most of VR and create mood and ambiance through other elements such as sound and lighting. And most importantly, you need to be able to save and share your vision.



(Fig. 2.3.1) Look back down at the menu at your feet. Find the lighting button and double tap it.

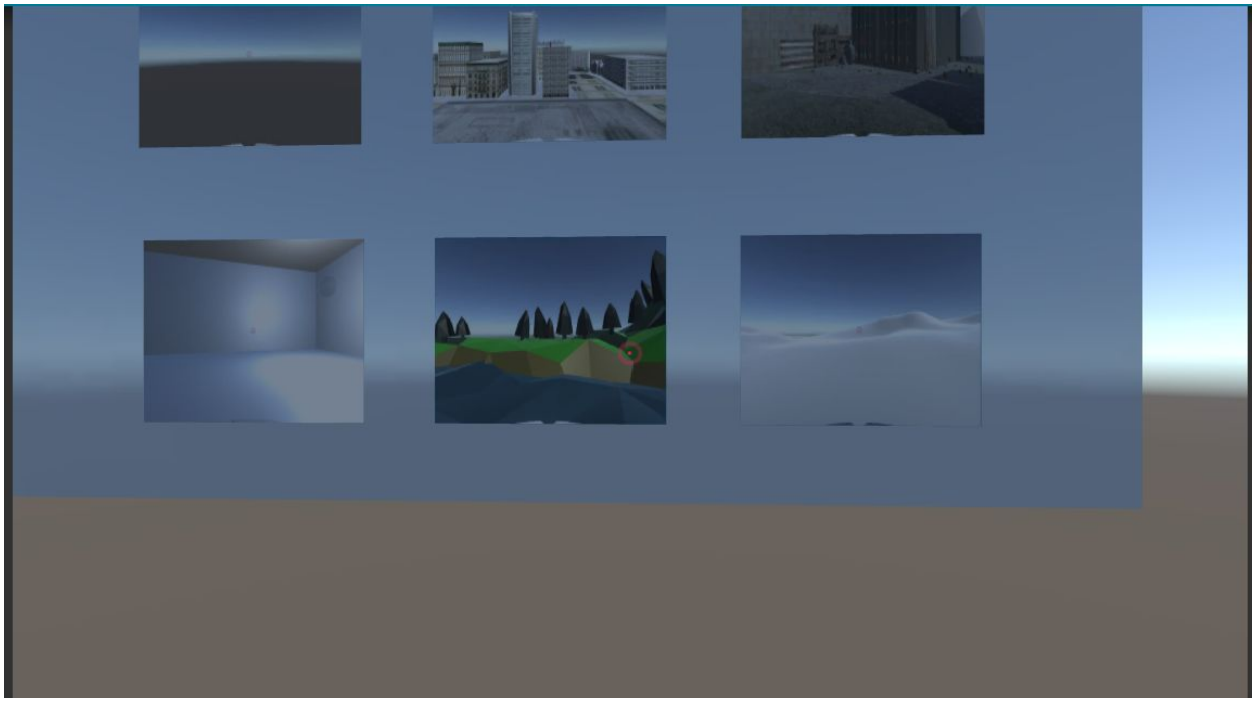


(Fig. 2.3.2) Look up to find the newly added light source. Double tap to reposition and change the brightness.

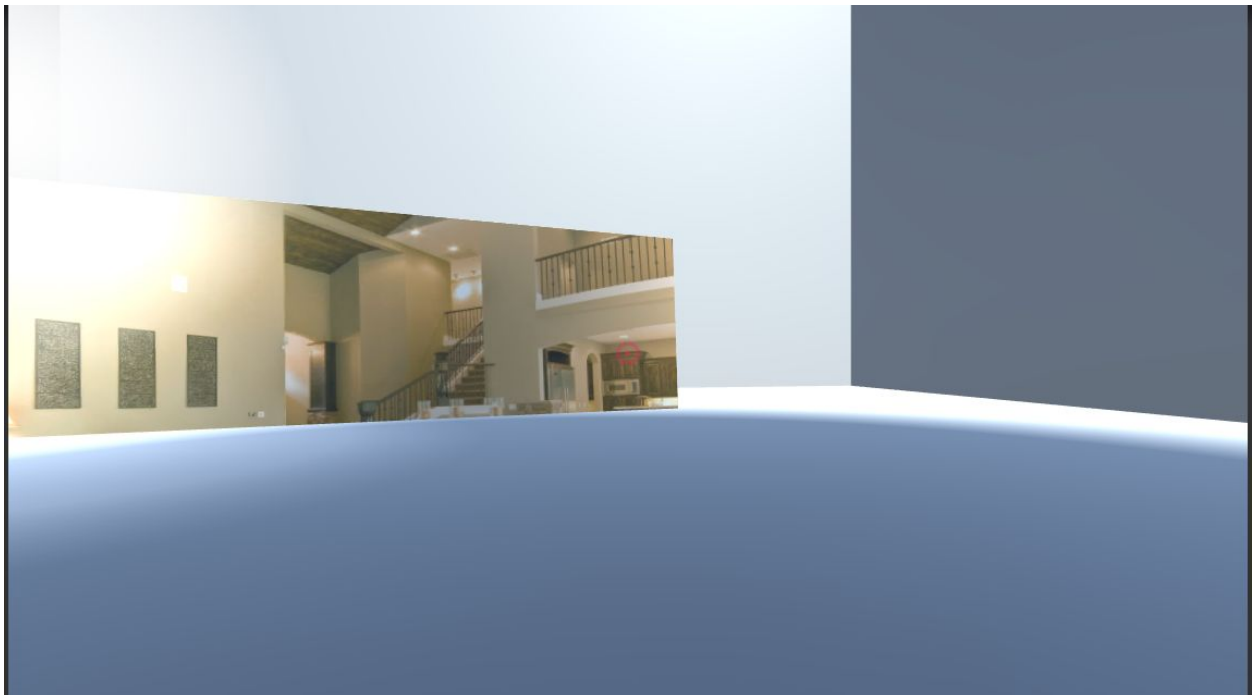


(Fig. 2.3.3) When you’re satisfied, double tap to position it. Lighting complete!

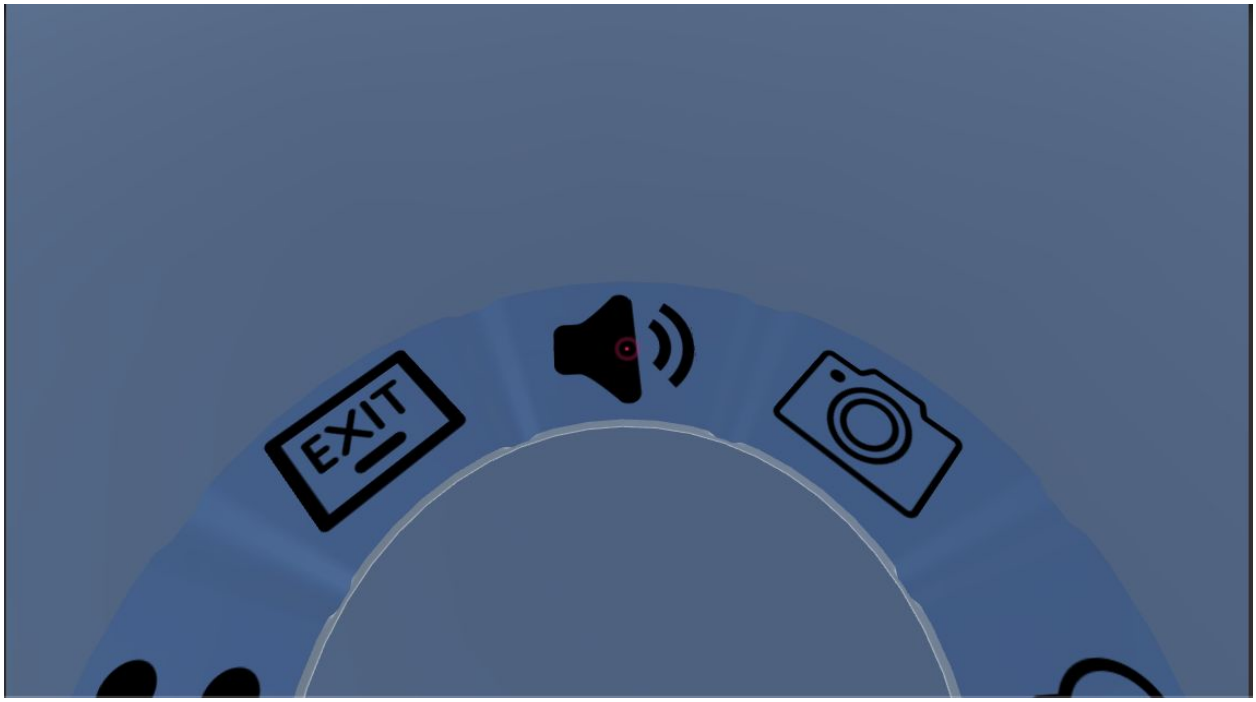
(Fig. 2.3.4) Look back down at your feet and find the terrain button. Double click to bring up the terrain panel.



(Fig. 2.3.5) Look up to find the terrain panel. Click on your terrain of choice.

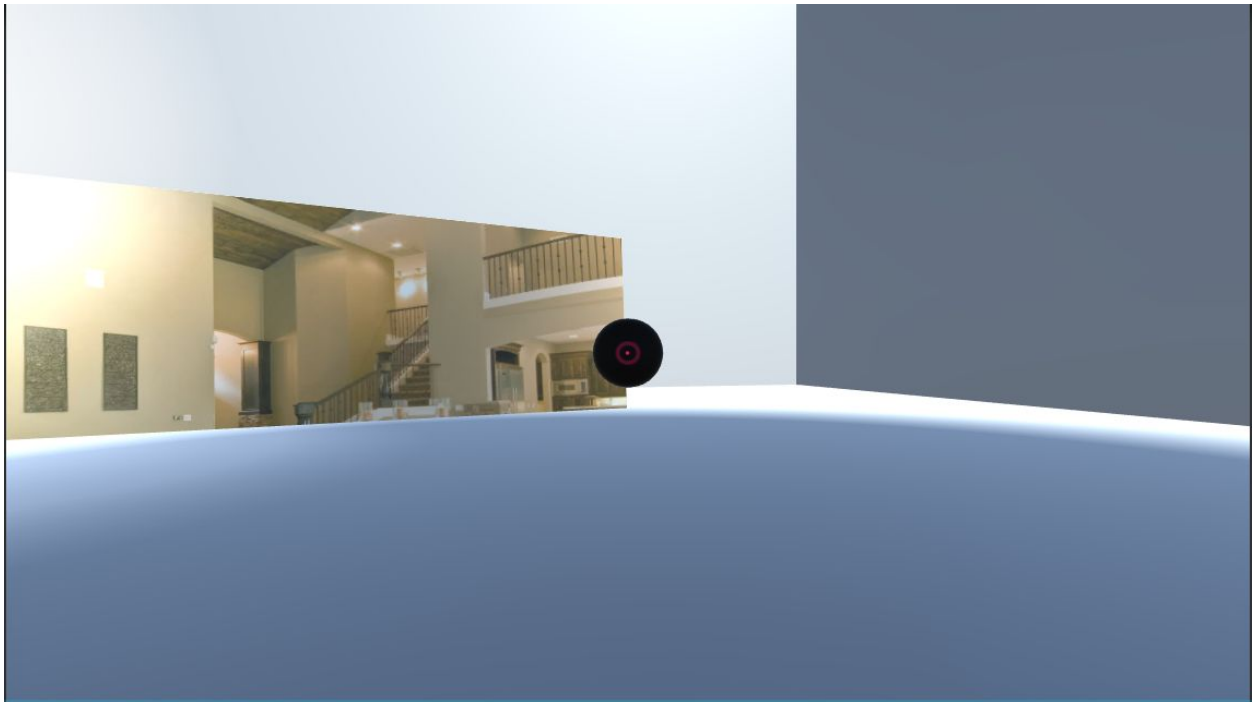


(Fig. 2.3.6) You should now find yourself in the terrain you selected along with all media you've positioned.



(Fig. 2.3.7) Look back down at the menu and find the audio button. Double click to bring up the audio panel.

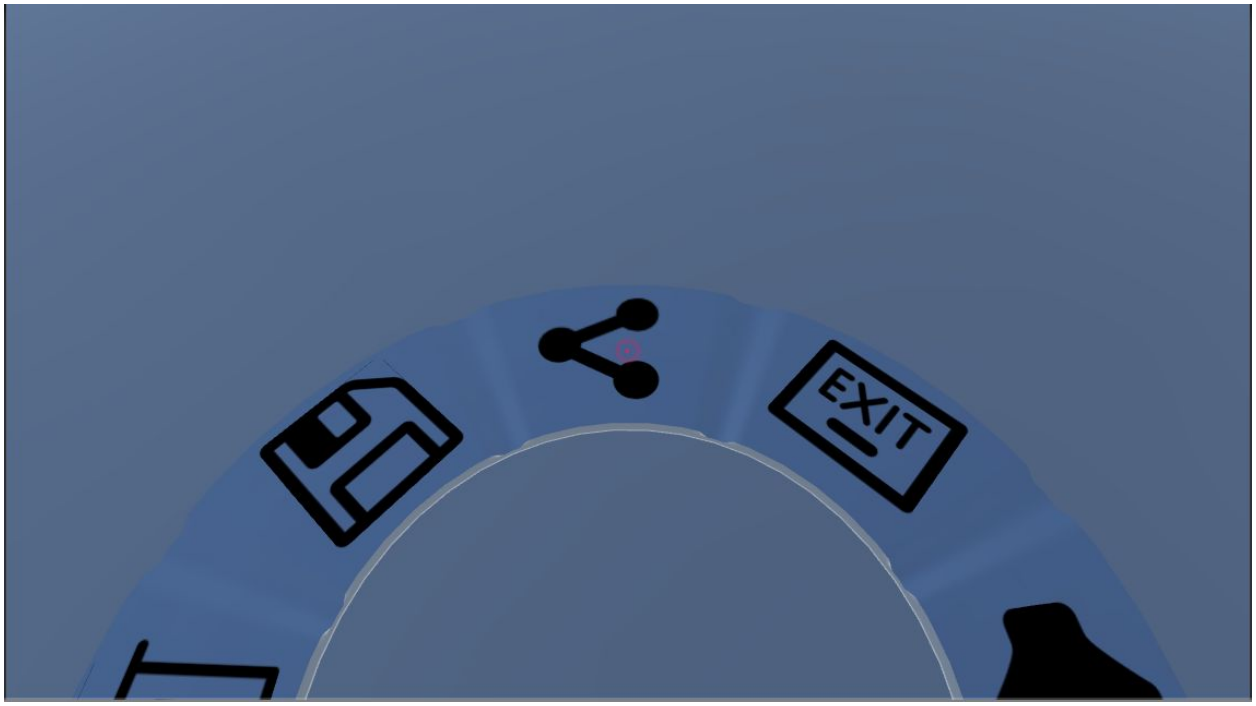
(Fig. 2.3.8) Look up and you should see the audio panel. Double click on the audio of your choice.



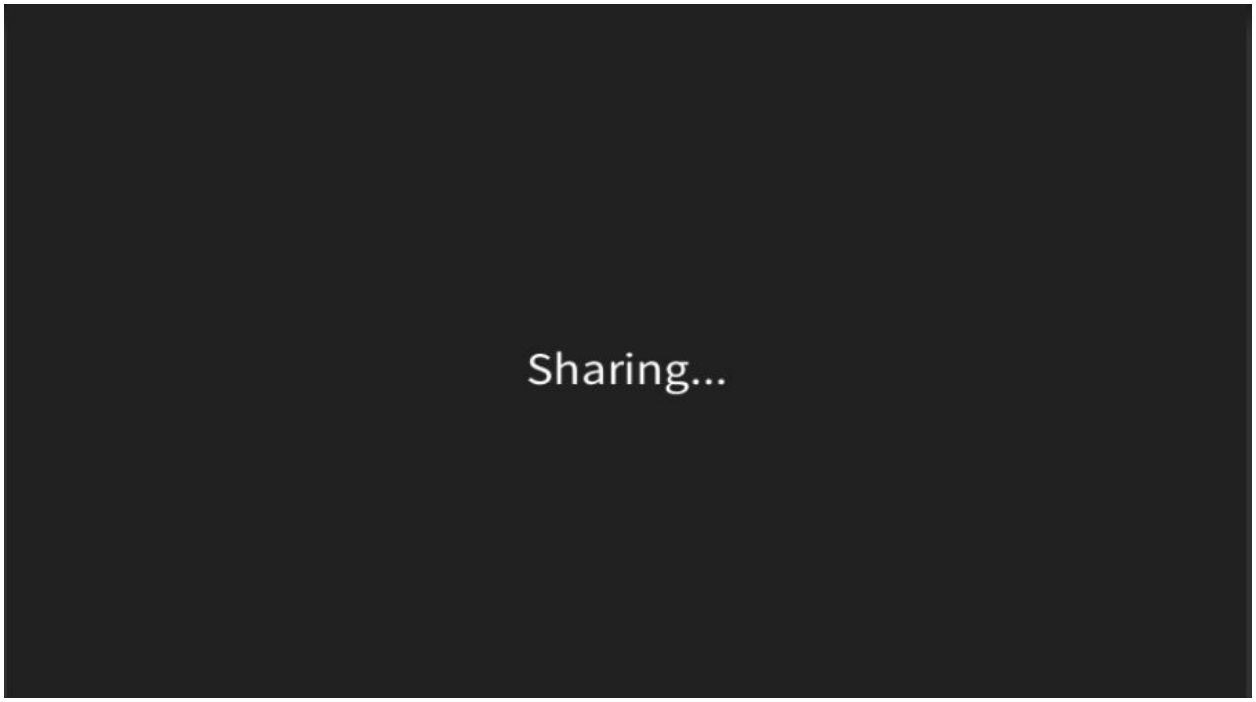
(Fig. 2.3.9) You should be able to see the audio sphere in your scene. Double click to reposition. Slide to adjust volume.

(Fig. 2.3.10) Now that you have elements in your scene, look down at the menu again and find the save button. Double click.

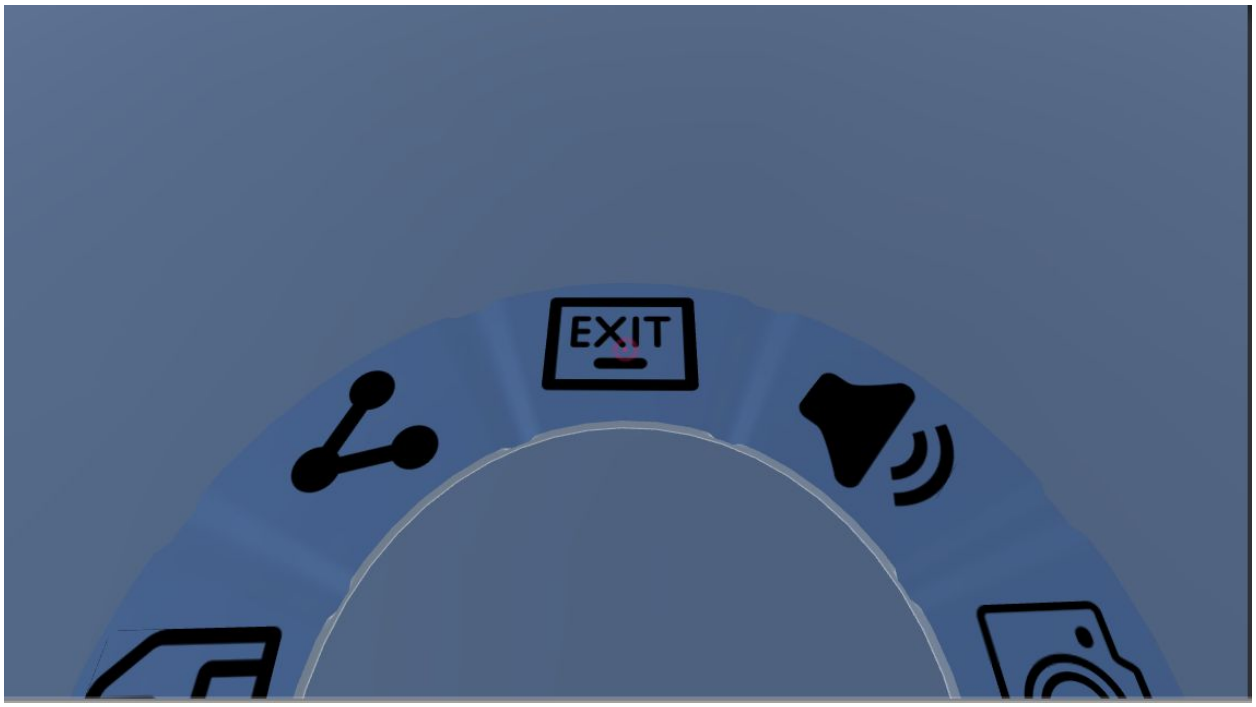
(Fig. 2.3.11) Saving. It will bring you back to your room when it has completed.



(Fig. 2.3.12) Now find the sharing icon. Double click.



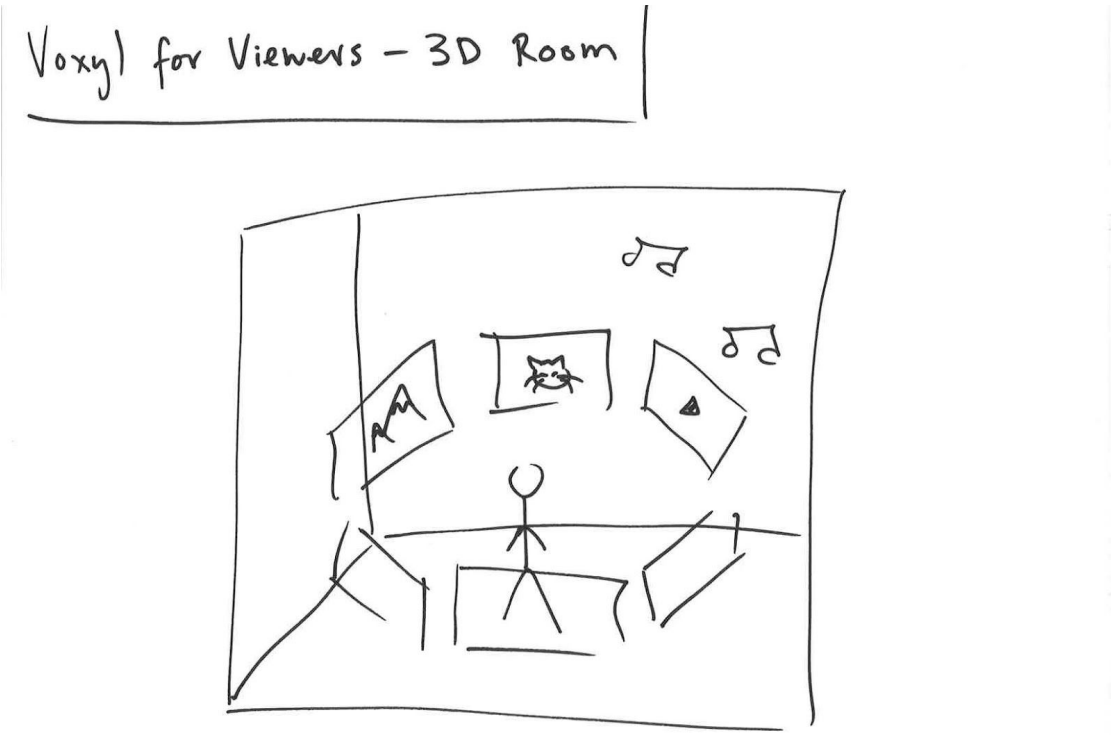
(Fig. 2.3.13) Sharing. When done, it will bring you back to your scene.



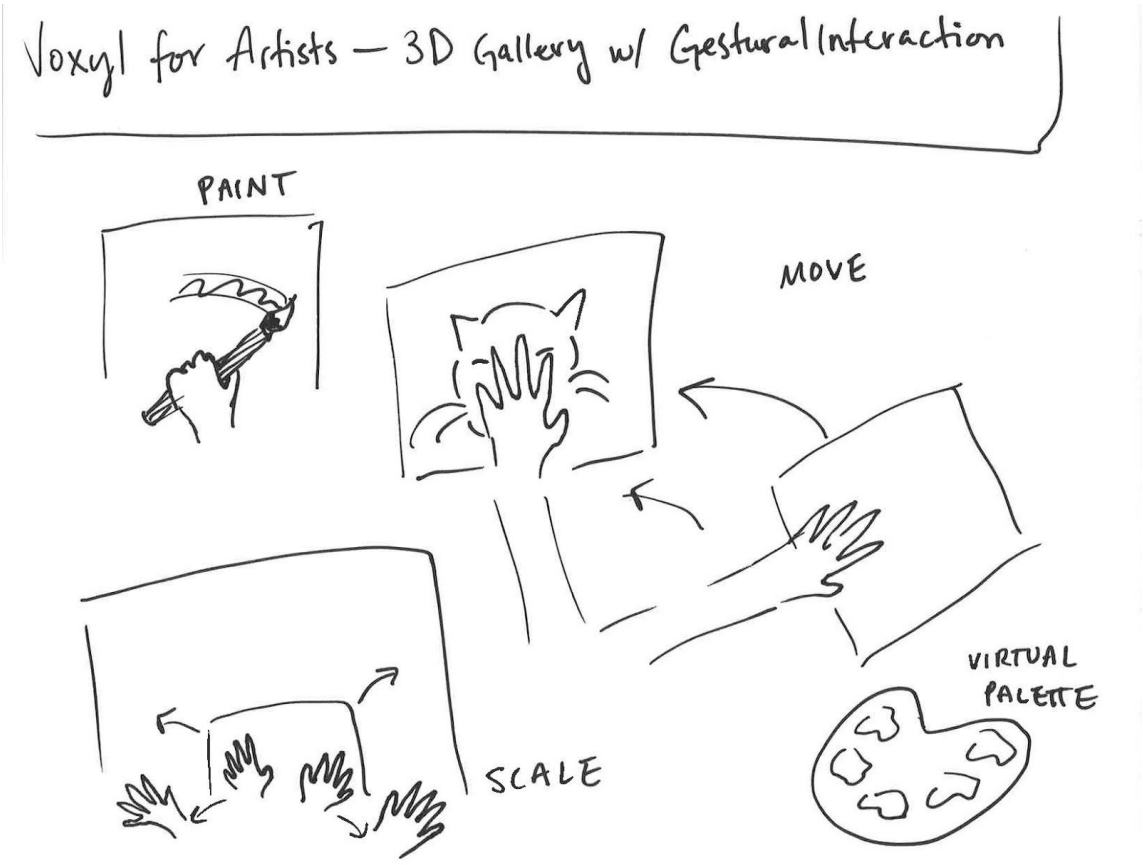
(Fig. 2.3.14) Now that you've saved and shared, find the exit button. Double click to be brought back to the home screen.

3. Design Evolution

Initial Sketches

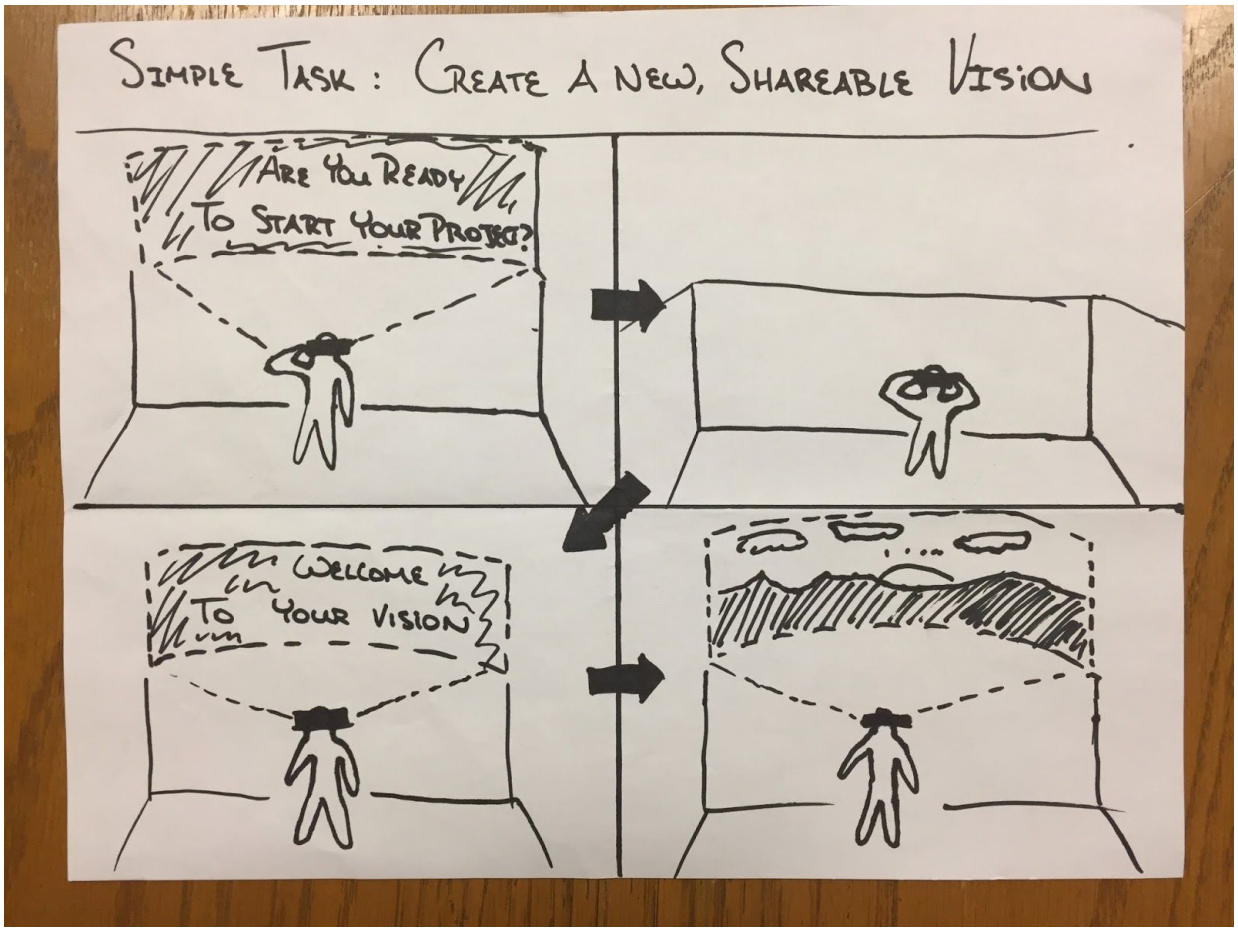


(Fig. 3.1) This sketch was one of three possible design ideas we came up with. This is the design idea that we ended up using.

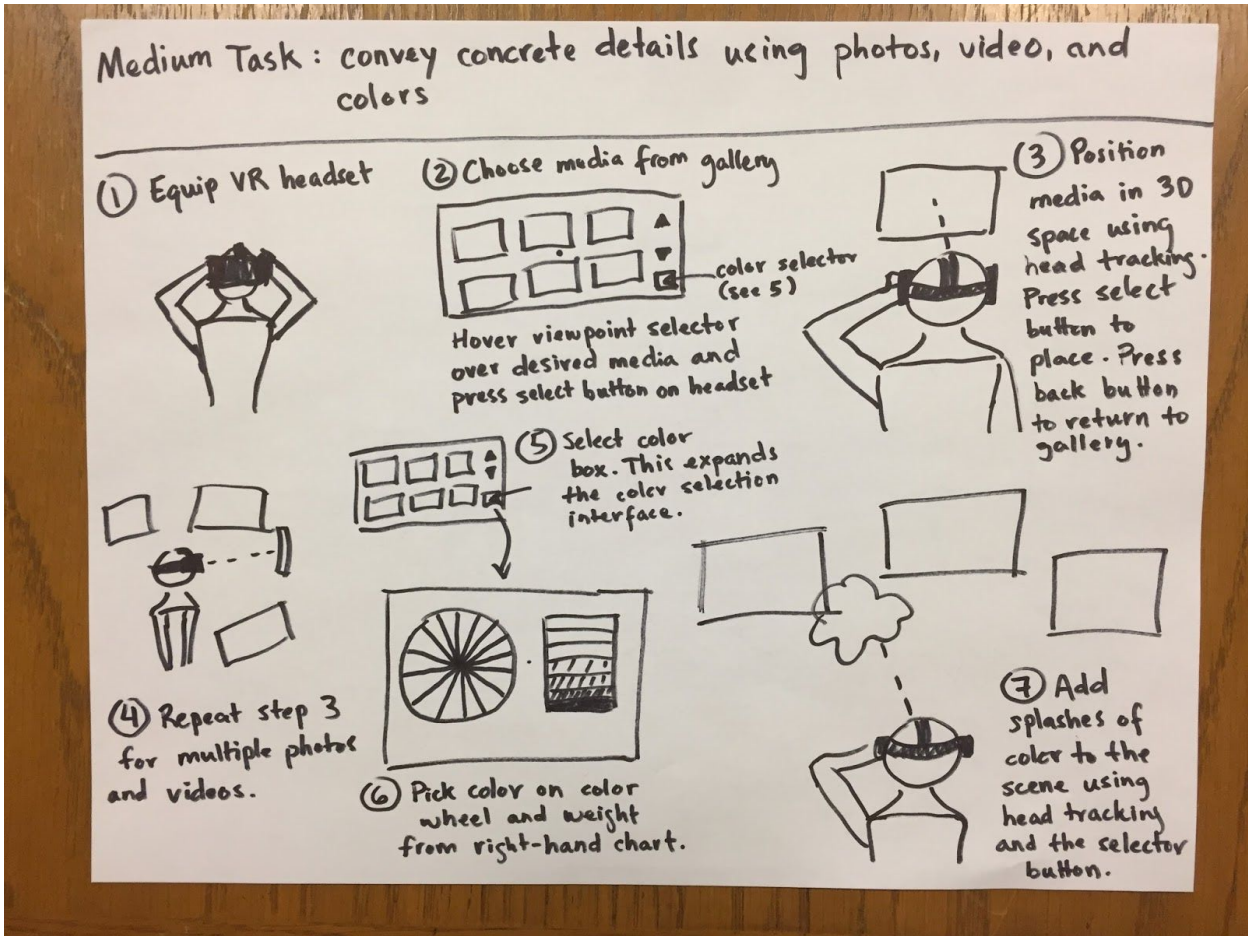


(Fig. 3.2) This is an expansion on the earlier 3D room sketch, illustrating how we originally intended to use gestural interaction to give the user control over their environment.

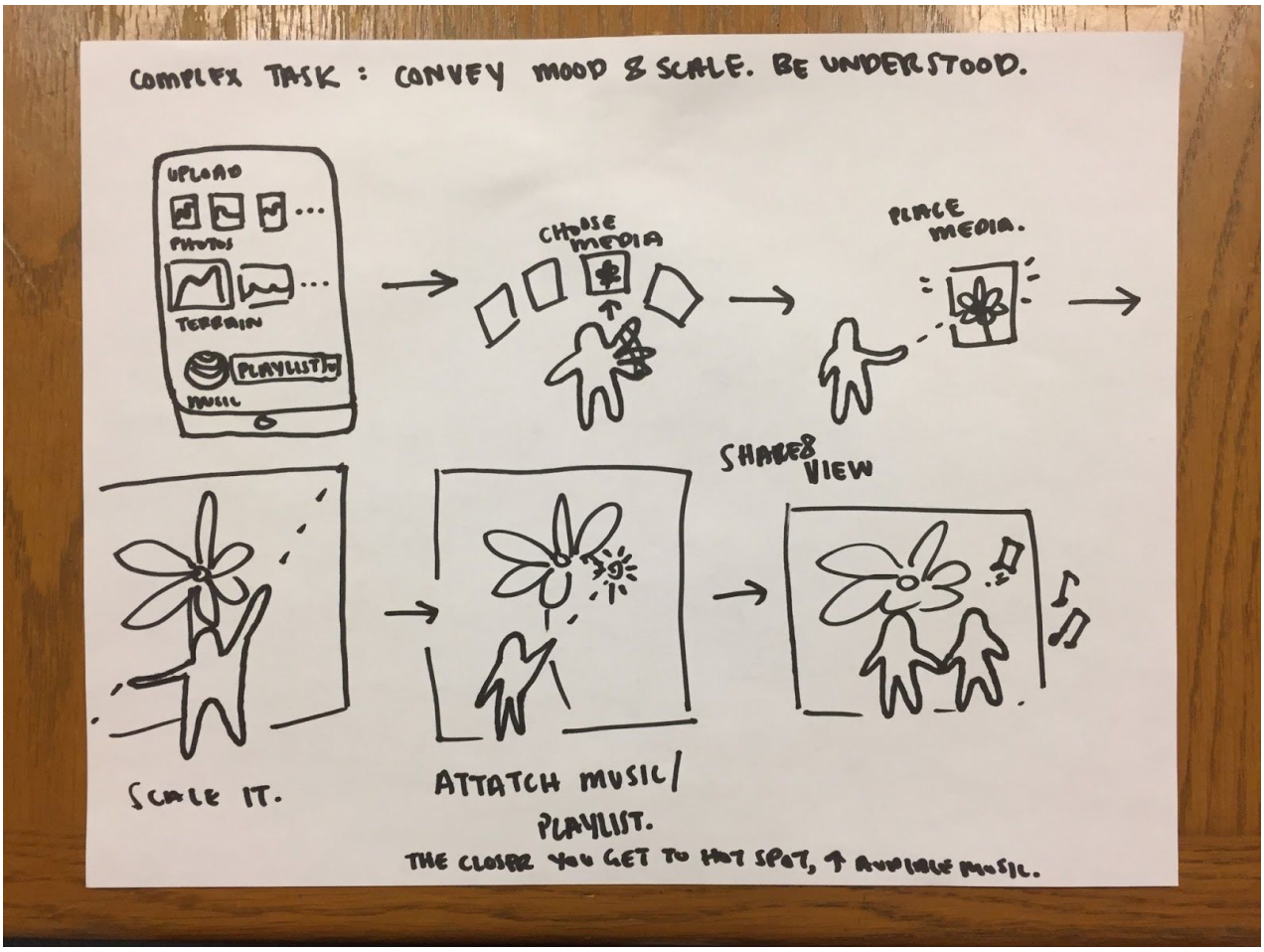
Low-Fi Prototype Storyboards



(Fig. 3.3) Storyboard for our first task: creating a new project.

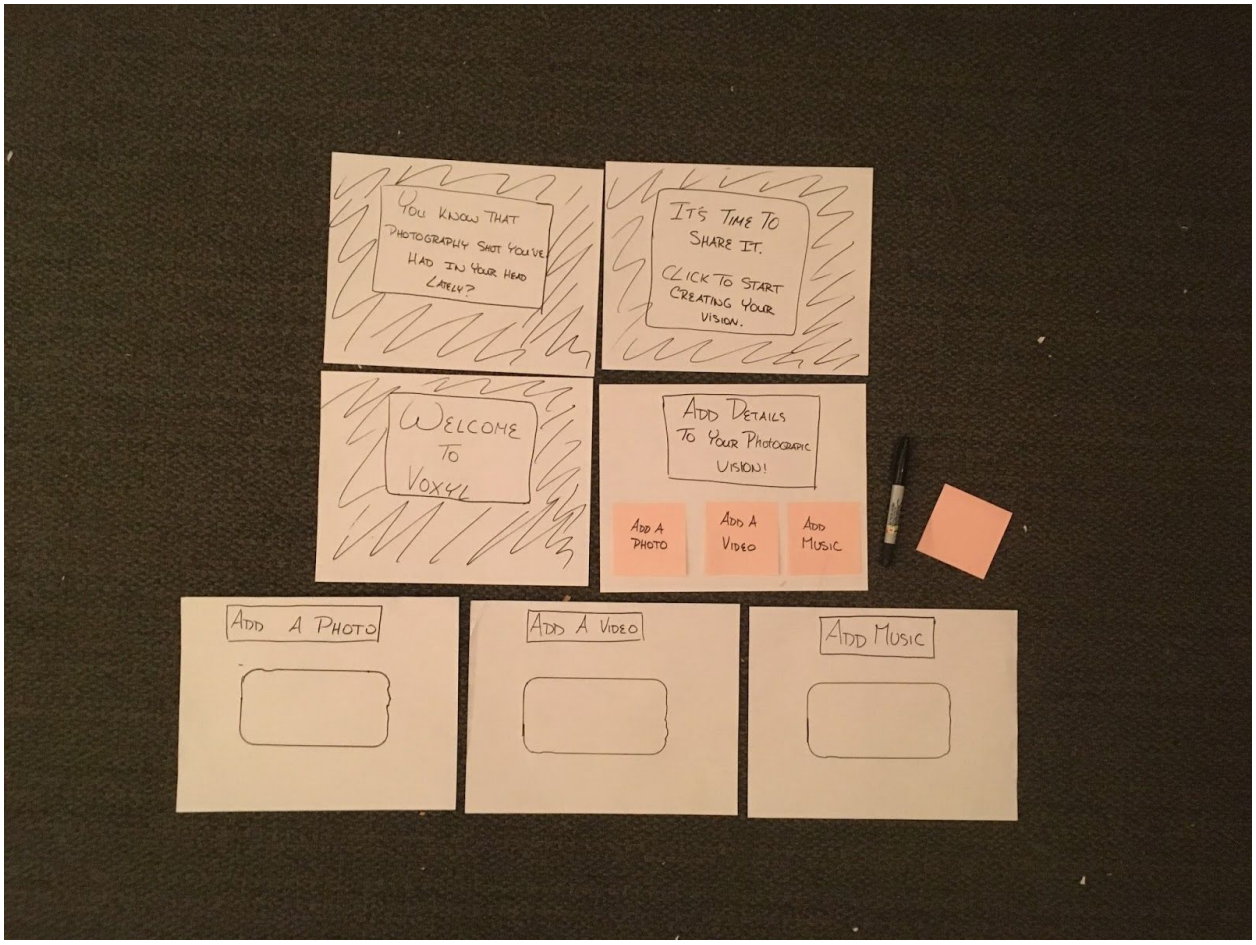


(Fig. 3.4) Storyboard for our second task, which changed between our low-fi and medium-fi prototypes. The original task, pictured here, would later be split between the second and third tasks.



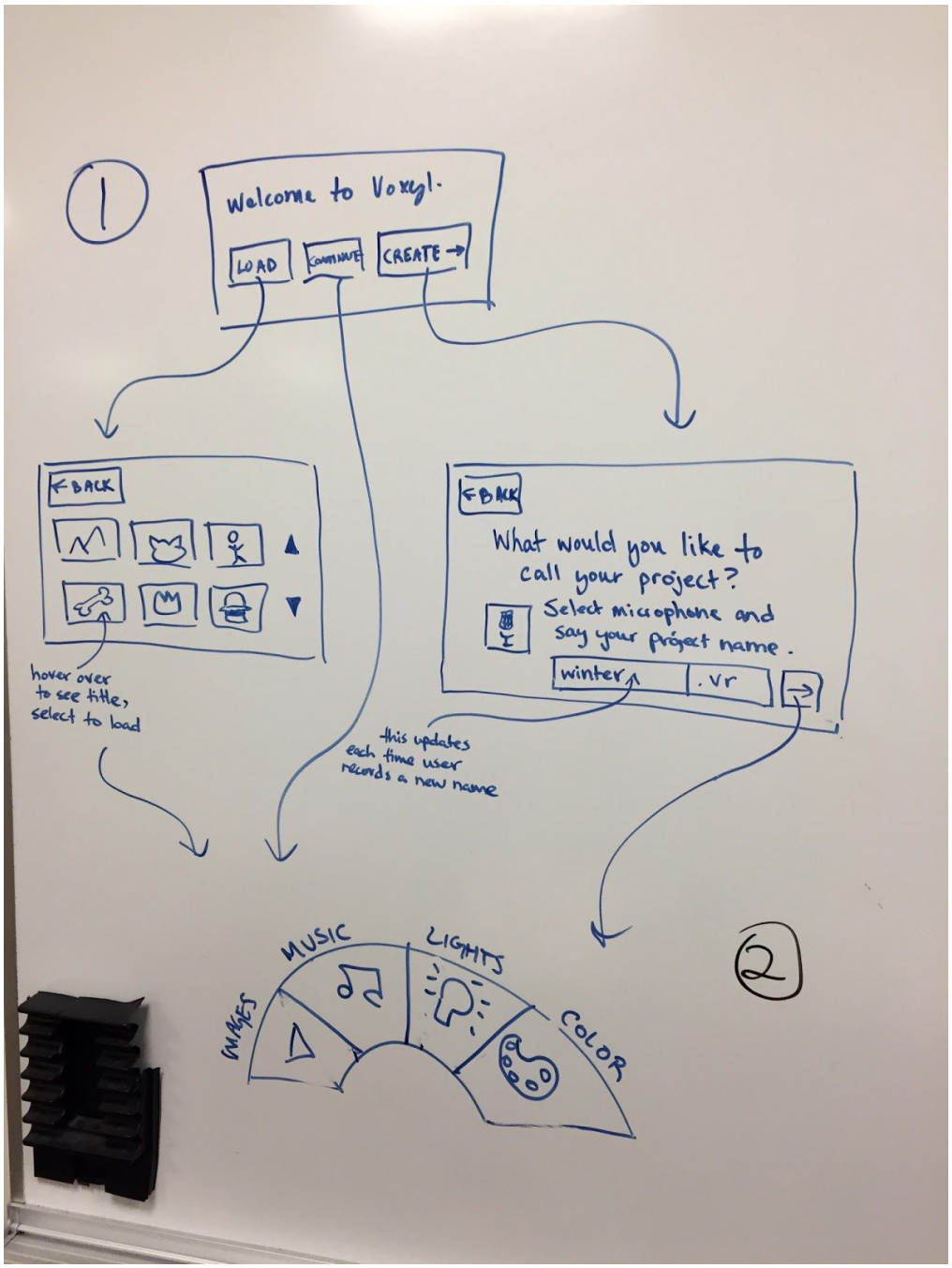
(Fig. 3.5) Storyboard for our third task, which would later be combined with elements of our second task to form a final third task.

Low-Fi Prototype

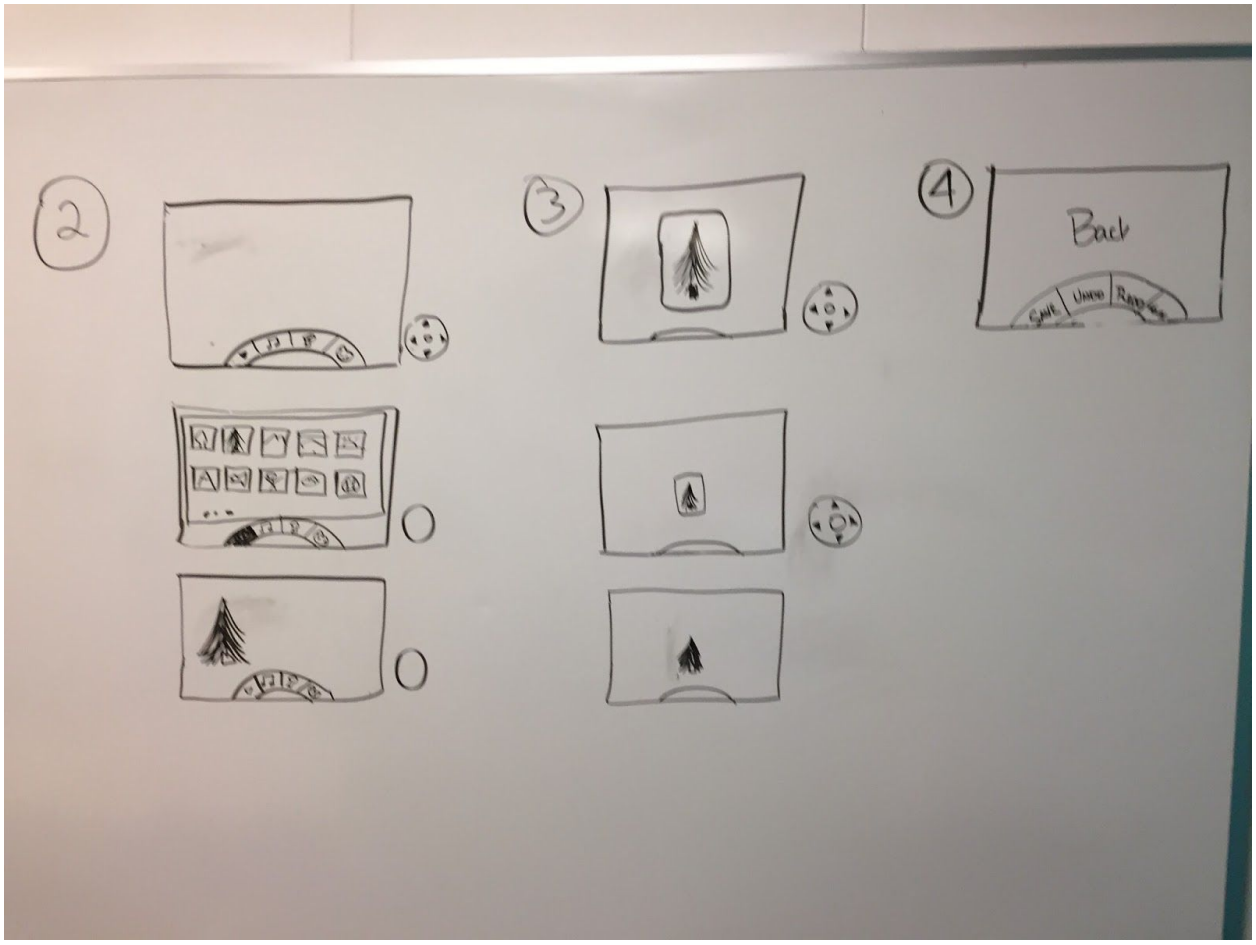


(Fig. 3.6) Our low-fi prototype. We tested this on participants by having a member of the team draw pictures that the user wanted to add to their 3D canvas. Adding music was done by playing the requested track on a laptop, and adding lighting was done by shining lamps on the canvas.

Medium-Fi Prototype Storyboarding



(Fig. 3.7) Storyboard of the first task for our medium-fi prototype. Here we began thinking about how we would give the user the ability to load previously saved projects, continue a recent project, and create and name a new project.



(Fig. 3.8) Storyboard of the second and third tasks for our medium-fi prototype. This is where the 360-degree radial menu was first conceptualized, along with the trackpad controls and popup menu with image gallery.

Medium-Fi Prototype

(Fig. 3.9) Welcome screen for our medium-fi prototype.

(Fig. 3.10) Empty canvas with 360-degree radial menu.

(Fig. 3.11) Popup menu with gallery of images to choose from.

(Fig. 3.12) Image after repositioning.

(Fig. 3.13) Placing a light source on the canvas.

(Fig. 3.14) Light source after placing.

(Fig. 3.15) Selecting a color to tint the canvas.

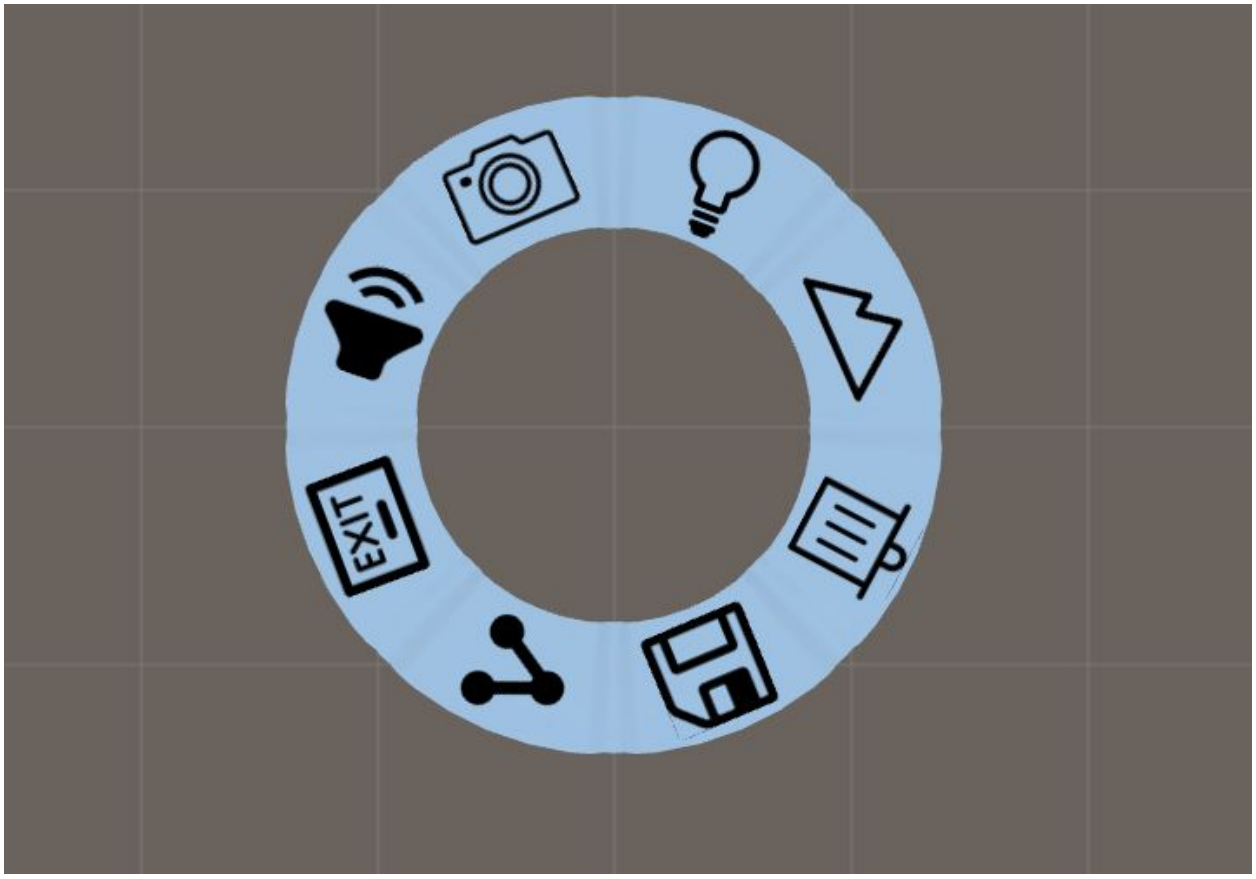
(Fig. 3.16) Tinted canvas after color selection.

(Fig. 3.17) Choosing an audio track to place on the canvas.

(Fig. 3.18) Placed audio track.

Hi-Fi Prototype

(Fig. 3.19) A scene created using our hi-fi prototype, viewed in Unity. Images, lighting sources, and audio have been placed in the scene and resized/repositioned. The 360-degree radial menu is visible in the center of the scene, below the main camera.



(Fig. 3.20) Top view of the 360-degree radial menu around the user's feet. Starting from the top and moving clockwise: add light, change terrain, trashcan, save, share, exit, add sound, add media.

4. Major Usability Problems Addressed

Heuristic Violations

Voxyl's heuristic evaluators reported one severe usability problem (severity 4) and four moderate usability problems (severity 3). We addressed all of these problems in our final prototype.

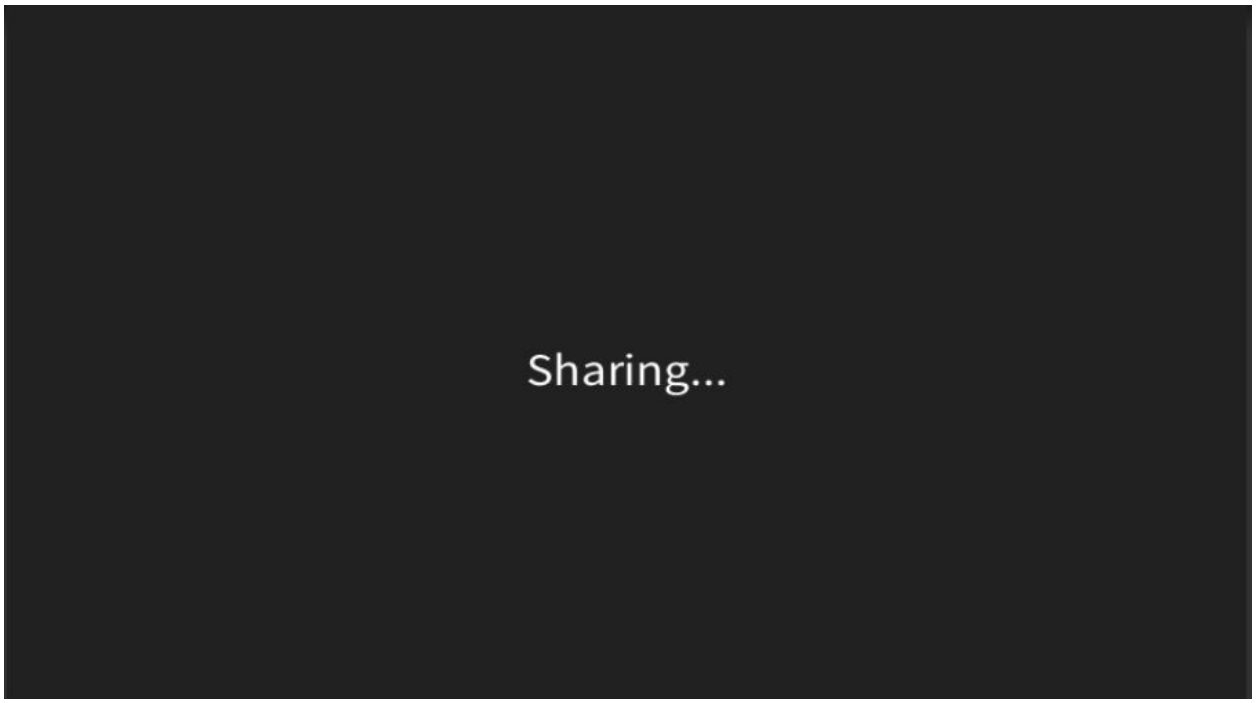
Save, Share, and Exit (Severity 4)

Violation: [H2-3: User Control & Freedom] There isn't any way to save or share the project, or to exit the project and go back to main menu screen. These options should be added to make the user experience more cohesive.

While the options to save, share, and exit the project had been part of our design plans all along, we had neglected to include a demonstration of those features in our medium-fi

prototype. Because our prototype included a 360-degree menu including 8 buttons, some of which were behind the user in virtual reality, we were not able to capture the full functionality of this menu in our medium-fi prototype, which was built to resemble a web application.

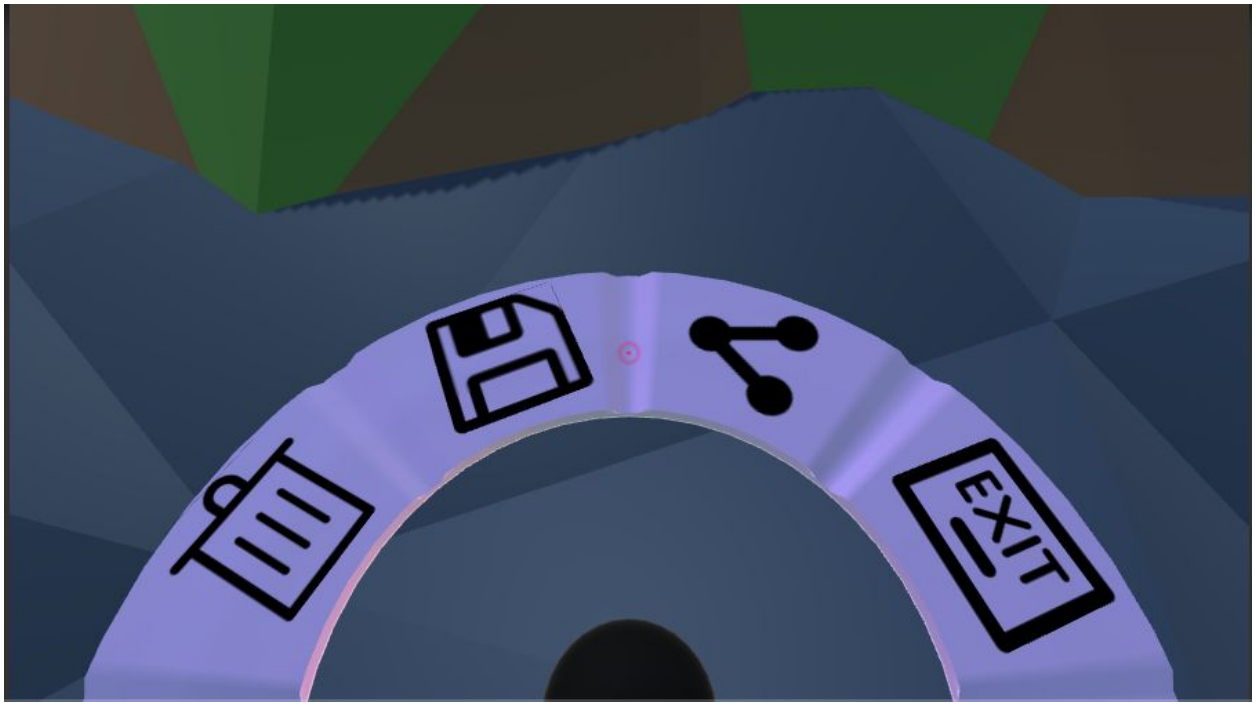
We included save, share, and exit buttons in this 360-degree menu in our final prototype (Fig. 3.20). The exit button returned the user immediately to the main menu, while the save and share buttons brought up a brief splash screen notifying the user that their creation had been saved/shared before bringing them back to the main menu (Fig. 4.1 and 4.2).



(Fig. 4.1) The splash screen that appears when the user selects the Share button on the menu.

(Fig. 4.2) The splash screen that appears when the user selects the Save button on the menu.

(Fig. 4.3) 360-degree radial menu as seen in virtual reality. Visible buttons: add sound, add media, add light, change terrain.



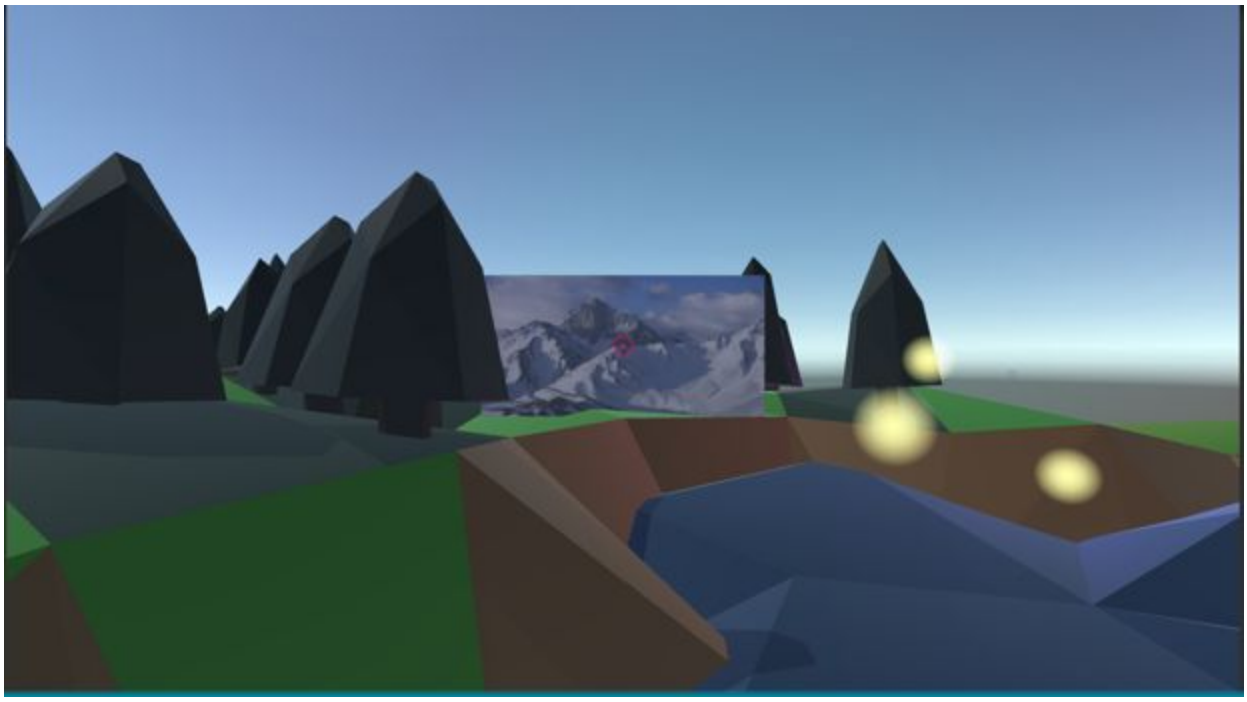
(Fig. 4.4) 360-degree radial menu as seen in virtual reality. Visible buttons: trashcan, save, share, exit.

Moving Pictures Forwards and Backwards (Severity 3)

Violation: [H2-2: Match Between System and Real World] In real life, pictures and objects are in a 3D space. To make the experience as realistic as possible,there should be the ability to move pictures forward and backward.

We agreed that this was an important feature to include in our final prototype, since the user would not be able to fully make use of the 3D space or juxtapose images on top of each other if we did not include this functionality. We addressed this violation by giving the user the ability to move selected objects toward or away from themselves by swiping left or right on the headset's touchpad while they had the object selected (Fig. 4.5 & 4.6).

(Fig. 4.5) A mountain image right after adding to the scene.

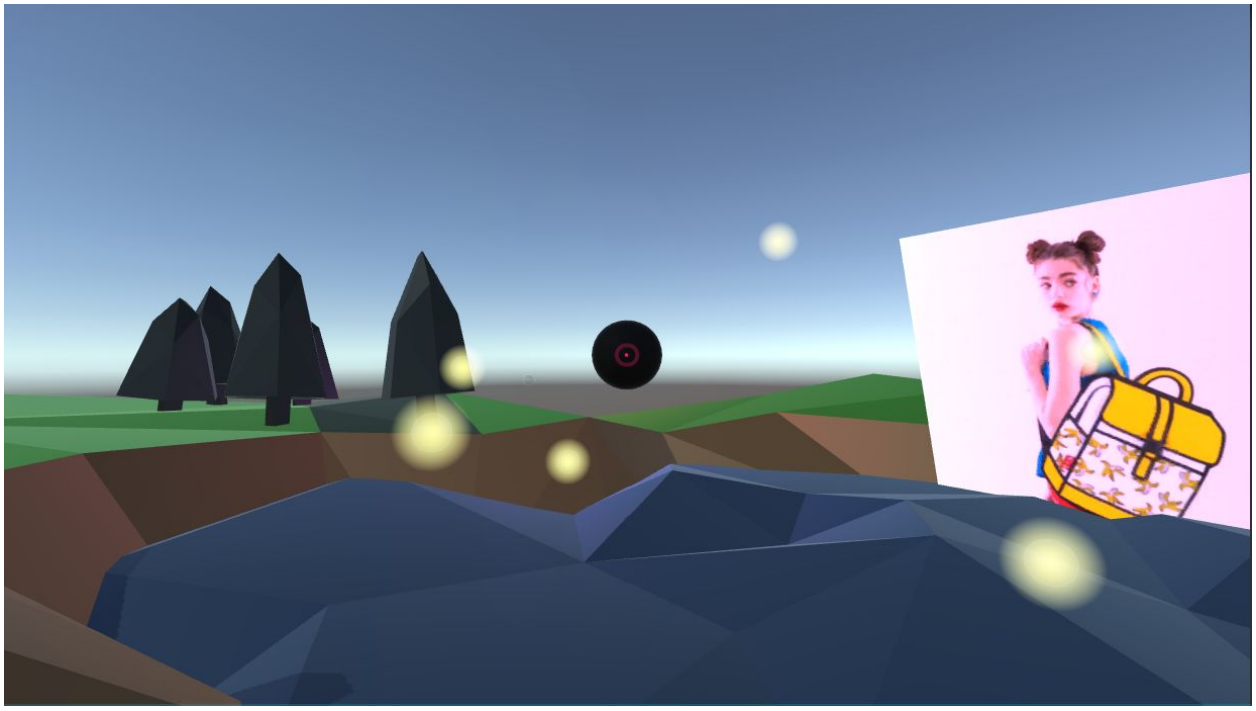


(Fig. 4.6) A mountain image after swiping left on the touchpad to move the image further away from the camera and swiping down to reduce the image size.

Changing the Volume of Audio Sources (Severity 3)

Violation: [H2-2: Match Between System and Real World] Since as described in the README the hi-fi prototype is going to allow adding multiple sources for music, there should be a way to control how loud different sources are to give a more natural feeling.

We had originally intended to only play audio sources when the user selected them or hovered over them, but in our final design we decided that it would give the user a more immersive experience if they were in an environment with multiple overlapping sources of 3D sound. It thus became very important to fix this violation. We did so by giving the user the ability to adjust the volume of any audio source by swiping up or down on the headset's touchpad while they had the source selected (Fig. 4.7). We also made the audio sources play 3D sound so that moving the audio source around the scene would also change the direction and distance that the sound played from.



(Fig. 4.7) Audio source selected in the center of the view. With this audio source selected, the user can swipe up/down on the touchpad to adjust its volume.

Making Lights Larger and Smaller

Violation: [H2-2: Match Between System and Real World] It is not clear how you would make a spotlight bigger or smaller, and what exact places it's going to light. For simplicity, you can allow changing the size, and the amount of light would be proportional to the size of the source.

We agreed that changing the intensity of light sources was important for making the scene feel more immersive. However, we decided against changing the size of light sources, since it didn't

feel natural for lights to grow larger or smaller in the scene. Instead we used the same functionality that we implemented for changing the volume of audio sources; swiping up or down on the touchpad with a light selected allowed the user to control the intensity of that light.

Undo/Redo and Removing Objects

Violation: [H2-3: User Control & Freedom] If the user makes a mistake there doesn't seem to be a way to undo or redo their actions. It would also be useful for a user to be able to delete images and light sources, and remove songs or colors. Adding these features to the menu would be pretty helpful for the user experience.

We agreed that removing objects from the scene was important functionality to include, in order to give the user greater control over their creation. However, we decided that undo/redo functionality was too complex for us to implement in such a short time span, and we settled for only implementing a deletion feature. This took the form of a trashcan icon on the menu (Fig. 4.4). The user can delete objects from the scene by selecting them and dragging them into the trashcan.

Other Changes

Our design underwent a lot of evolution once we started programming in Unity and began to understand the limitations of our hardware. The user's interaction with the scene is limited by the number of controls available to them: tapping the touchpad, swiping on the touchpad, and using the back/menu buttons on the headset.

Unity itself also imposed some limitations on our design. For example, in order for our drag-and-drop scripts to work on non-image objects like audio and light sources, we had to make audio and light manifest as physical objects in the scene. We chose small spheres--shiny black for audio (Fig --) and transparent for lighting--to represent these objects. If we had had more time, we would have implemented a show/hide toggle for these objects.

We also ran out of time to implement the "add color" feature, which proved to be more complex than adding other types of objects to the scene. In order to add color to the scene, the user needs to specify two variables: the color that they want and the object that they want to tint. Due to limitations in the number of controls available to the user while they have an object selected, we would have been forced to explore some fairly complex avenues of implementation for this feature, which we ended up not having time for.

5. Prototype Implementation

We built our prototype in Unity. The final product is designed to be run using a mobile phone (Samsung Galaxy) and a virtual reality headset (Samsung Gear VR).

Unity was challenging to work with because most of our team did not have any prior experience with Unity or C#, the language used to write scripts in Unity. Unity is a program primarily used for making 3D or 2D mobile games; an application built with Unity consists of scenes containing objects. Scripts are attached to objects and are event-based, with different callback functions for when an object is initialized, updated, clicked on, deleted, and so on.

Some of our features were much easier to implement in Unity than others. Attaching scripts to objects that fire when the object is clicked on--such as our drag and drop selection script--were fairly easy to implement. However, anything to do with object collisions--such as trying to detect whether an object is currently colliding with another for purposes of trashcan deletion or using a virtual paintbrush to color objects--is difficult and non-intuitive to program in Unity. We ended up having to hardcode in the position of the trashcan icon on the menu and check to see whether each object was overlapping with those coordinates for the purposes of collision detection.

We primarily used Wizard of Oz techniques when handling file I/O. The splash screens that appear when the user selects Save or Share from the menu indicate that the user's changes are being saved or shared, but nothing is actually happening under the hood. Similarly, selecting Load from the main menu only loads a single premade scene instead of scenes that the user has previously saved.

We used hard-coded data for the lists of images and audio that can be added to each scene, instead of pulling from images and audio on the user's phone as advertised. We also hard-coded in a selection of virtual environments instead of giving the user the ability to import 3D terrain files.

The final prototype is missing only a few features that we originally intended to implement: the ability to name a new project during project creation, the ability to add colors to the canvas, and the ability to add videos as well as still images. We simply did not have time to implement these features for the prototype, but if we continued working on this project in the future, we would definitely want to include them. Other features on our wishlist include the ability to rotate objects, change the color of lighting, add more 3D objects, and change the weather and time of day in the virtual environment.

6. Summary

Voxyl was an immensely satisfying project for us to work on. We were able to explore a unique problem space and come up with a solution using immersive virtual reality. Since we picked a technology that none of us had designed for before (Samsung Gear VR), we were all able to learn new skills and produce something that was as much experiment as solution. Our design underwent a great deal of evolution throughout the process, particularly between the low-fi and medium-fi prototypes, but we were able to capture the essence of our solution from beginning to end: giving artists ultimate control over a 3D space to convey details of their creative vision.