**CS4620 | CS5620**

Fall 2024

## Final Project

# 🖼️ EXPLORE THE WONDERS OF A. DAVIS ART MUSEUM 🖼️

JOANNA YOO (SY797), LIAN LIAO (LL987), ANDREW YANG (ACY46), ZHENG WANG (ZW724)

# Table of Contents

Contribution Map (Summary) 1

Project Overview & Design/Development Process 2

Core Gameplay Mechanics 3

Technical Features 4

How to Run & Asset List 5

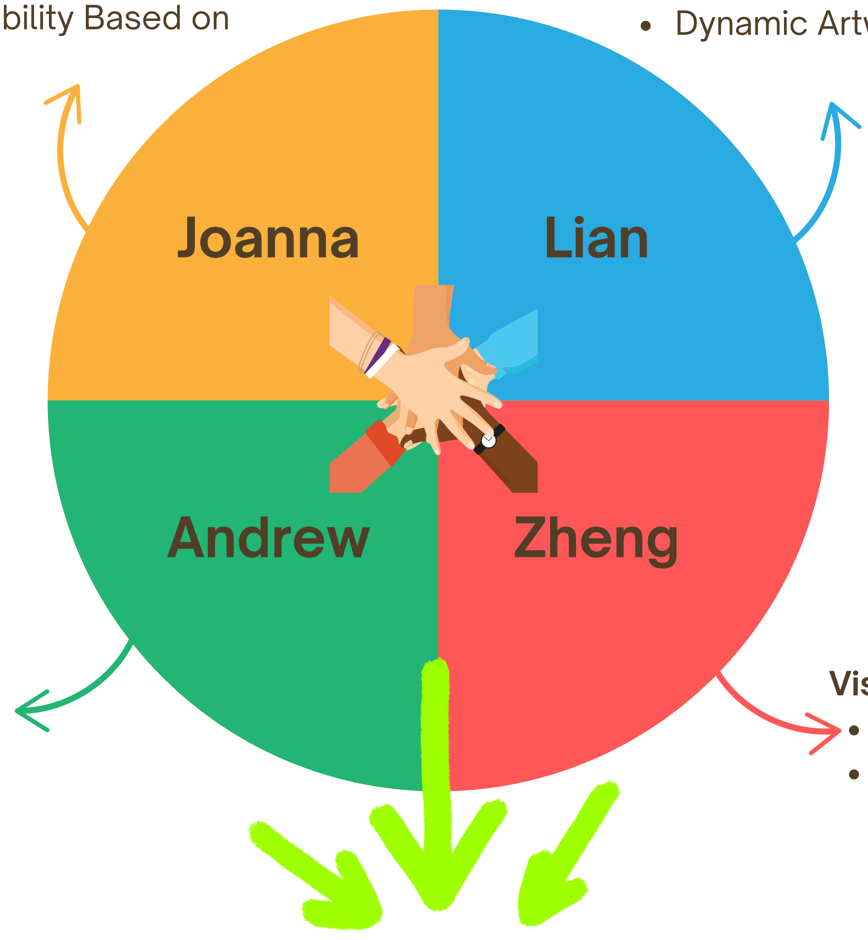
# Summary (Contribution Map)

**Character Control/ Camera Movement**
- Character Rotation
- Camera Zoom/Movement

**Object/Character Interaction**
- Artwork Visibility Based on Proximity

**Character Control/ Camera Movement**
- Smooth Character Movement
- Camera Follow

**Object/Character Interaction**
- Dynamic Artwork Descriptions



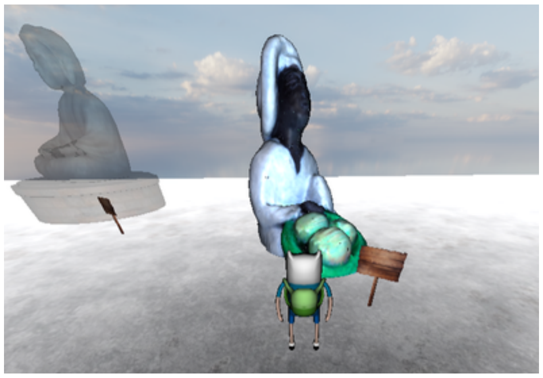
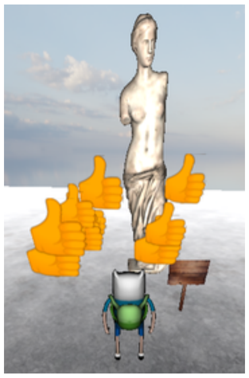
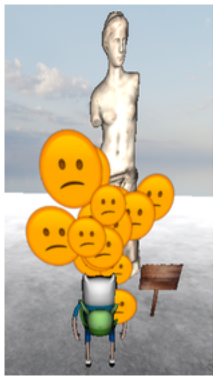
**Particle Systems**
- User Reaction

**Visual Enhancement**
- Terrain texture
- Skyview

## Together

- Collaborated on concept development, ensuring a unified vision for the project.
- Engaged in constant debugging and made minor improvements across various features.
- Provided support to each other when stuck on challenging issues.
- Attended all meetings promptly, contributing to regular progress updates and planning.

## Project Overview

Our team developed a virtual museum simulation where Finn, the character from Adventure Time, explores various artworks on the map. The simulation includes features such as artwork becoming visible only when within range, similar to how objects in real life come into view as you approach them. Players can also see artwork information when near a piece, much like reading placards in an actual museum. Additional details include a particle system to express the player's reaction to artworks, textures and shaders designed to mimic the holistic vibe of a museum, and smooth player interactions achieved through interpolation. These elements combine to create an engaging and immersive museum experience.

## Design & Development Process

**Sprint Planning** → **Sprint 1** *Camera Control & Character Movement* → **Sprint 2** *Object & Character Interaction* → **Sprint 3** *Particle System & Visual Enhancement* → **Sprint 4** *Testing & Final Refinements*

Sprint Planning: Defined the game concept, core mechanics, and scope

Sprint 1: Implemented camera control and character movement features

Sprint 2: Implemented features for displaying artwork descriptions and added transparency effects

Sprint 3: Added particle effects for player's reactions and worked on terrain shaders and the skymap.

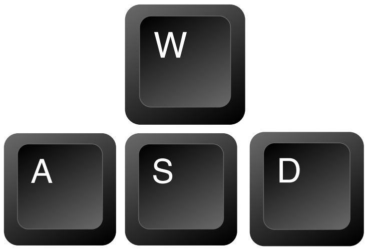
Sprint 4: Gathered feedback from teammates, polished the game, and resolved bugs to ensure a seamless and engaging user experience.

Our team followed the **Agile methodology** for the project design and development process. In each sprint, we held regular meetings to incorporate feedback and ensure that each phase focused on incremental improvements. Continuous refinement was a key aspect of our approach, allowing us to build and enhance the game step by step. We also emphasized collaboration across team members to ensure that all features, from character movement mechanics to visual details, were well-integrated and met the project objectives. This iterative process allowed us to maintain flexibility and deliver a polished final product.

## Core Movement Mechanics

**(Note: All key inputs are case-insensitive.)**

### Keyboard Components



**W and S:** Move the user forward and backward
**A and D:** Rotate the user left and right

**R**- Zoom in

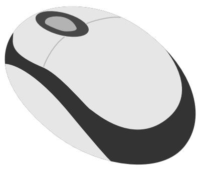
**F-** Zoom out

**U-** 👍 Thumbs Up Reaction

**I-** 😍 “I love it!” Reaction

**O-** 🙁 **“**I don’t like this” Reaction

### Mouse Control



The user can control the camera view by dragging the mouse.

## Technical Features

### Character Control/ Camera Movement (Joanna & Lian)

- **Character Rotation:** Added functionality to rotate the character left or right using 4D matrix transformations, enhancing navigation and control. Updates included revising the HasPosition3D interface to include attributes like orientation and targetPosition in addition to position. (Joanna)
- **Smooth Movement:** Utilized interpolation to ensure the character's movement transitions are smooth, avoiding any choppiness (Lian)
- **Camera Zoom and Movement:** Integrated a basic zoom feature along with camera movement controls for an enhanced player view (Joanna)
- **Camera Follow:** Implemented a dynamic camera system that tracks and follows the character during movement (Lian)

### Object/Character Interaction (Joanna & Lian)

- **Dynamic Artwork Descriptions**: When the character moves within a specified range of an artwork, a corresponding description dynamically appears at the top of the UI screen, providing detailed information about the artwork (Lian)
- **Artwork Visibility Based on Proximity:** All artworks are transparent by default, enhancing focus on the environment. An artwork becomes opaque and fully visible only when the character is within the specified proximity range, creating an interactive and immersive experience. The opacity control was custom-developed and integrated into the existing **demoshader.glsl**, dynamically adjusting transparency based on the character's distance. (Joanna)

### Particle Systems (Andrew)

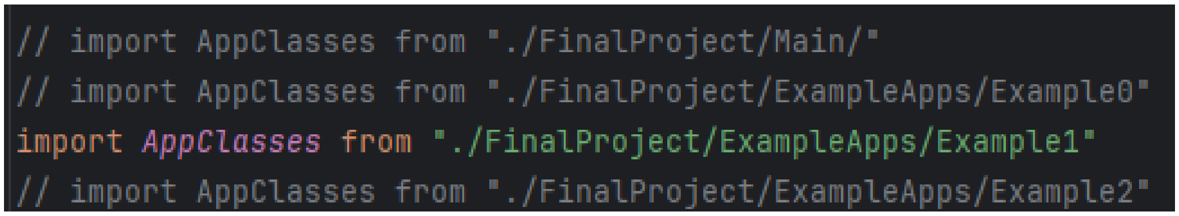
- **User Reaction:** The particle systems were implemented to respond dynamically to user reactions. Each interaction triggers unique properties, including varying speeds, shapes, and behaviors, creating an engaging and visually immersive experience tailored to the user's input.

### Visual Enhancement (Zheng)

- **Terrain Texture:** The terrain texture was modified to a mirror-like surface using enhancements in the **terrain.glsl** shader. By integrating the Blinn-Phong lighting model with reflection and Fresnel effects, the floor dynamically responds to light and viewing angles. Normal mapping adds subtle surface detail, while customizable parameters like brightness and reflection intensity allow for fine-tuning. This creates a refined, realistic environment perfect for a museum setting
- **Skyview:** The HDRI was converted to a cubemap format for precise six-sided texturing (px, nx, py, ny, pz, nz), ensuring a seamless sky environment. Exposure levels were carefully adjusted to harmonize with the reflective floor and overall lighting, creating a balanced and cohesive visual experience.

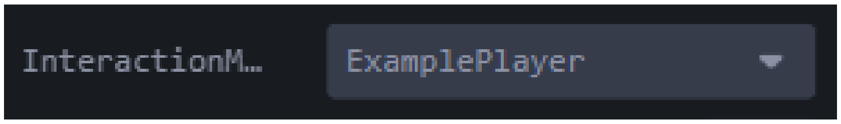
## How to Run

1. Ensure that the project is set up with the correct import for the example base. In MainApp.tsx, uncomment the following line:
import AppClasses from "./FinalProject/ExampleApps/Example1";



```
// import AppClasses from "./FinalProject/Main/"
// import AppClasses from "./FinalProject/ExampleApps/Example0"
import AppClasses from "./FinalProject/ExampleApps/Example1"
// import AppClasses from "./FinalProject/ExampleApps/Example2"
```

2. Verify that the interactionMode is set to "ExamplePlayer" in the relevant configuration.



```
InteractionM…    ExamplePlayer
```

These steps ensure the application runs correctly based on our implementation. 😊

Most of our implementations are located in the **FinalProject-StarterCode** and **FinalProject-Example1** directories. Key features are also implemented in shader files such as demoshader.glsl and terrain.glsl, as referenced on page 4. Dynamic interaction logic is primarily handled in the **GUIHelpers.tsx** file. While there may be a few additional functions or minor components outside these locations, the majority of the work can be found in the mentioned files and directories.

## Asset List

**Finn Character**
https://www.turbosquid.com/3d-models/finn-the-human-1296287

**Nubian Woman Sculpture**
https://www.turbosquid.com/3d-models/handmade-nubian-figurine-3d-model-1835192

**Venus de Milo**
https://free3d.com/3d-model/statue-v1--541832.html

**The Thinker**
https://free3d.com/3d-model/the-thinker-v3--850984.html

**Meditating Buddha**
https://free3d.com/3d-model/statue-v1--856385.html