**Technical University of Crete**
**School of Electrical and Computer Engineering**

# Diploma Thesis

## AI-Powered Digital Puppet Creation and Collaborative Shadow Theater Integration

**Christos N. Ioannidis**

**Supervisor**
Professor Aikaterini Mania (ECE)

**Thesis Committee**
Professor Aikaterini Mania (ECE)
Professor Michail G. Lagoudakis (ECE)
Professor Antonios Deligiannakis (ECE)

**February 2025**

**Πολυτεχνείο Κρήτης**
**Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών**

# Διπλωματική Εργασία

## Δημιουργία Ψηφιακών Μαριονετών με Υποβοήθηση Τεχνητής Νοημοσύνης και Ενσωμάτωσή τους σε Συνεργατικό Θέατρο Σκιών

**Χρήστος Ν. Ιωαννίδης**

**Επιβλέπουσα**
Καθηγήτρια Αικατερίνη Μανιά (ΗΜΜΥ)

**Τριμελής Επιτροπή**
Καθηγήτρια Αικατερίνη Μανιά (ΗΜΜΥ)
Καθηγητής Μιχαήλ Γ. Λαγουδάκης (ΗΜΜΥ)
Καθηγητής Αντώνιος Δεληγιαννάκης (ΗΜΜΥ)

**Φεβρουάριος 2025**

# Abstract

This diploma thesis introduces significant advancements to the digital storytelling platform eShadow [8] by integrating Artificial Intelligence (AI)-powered puppet creation workflows and real-time collaborative features. These enhancements address limitations in accessibility, usability, and multi-user interactivity, broadening the platform's applicability in educational and artistic settings.

Leveraging state-of-the-art machine learning and computer vision techniques, the proposed system transforms diverse inputs—ranging from real-world objects to hand-drawn images—into digital puppets, overcoming constraints in previous approaches. The solution integrates high-resolution image processing, automated background removal, and generative AI to create aesthetically cohesive puppets with minimal user intervention. A fine-tuned LoRA (Low-Rank Adaptation) model is employed to replicate the traditional shadow theater art style, ensuring stylistic authenticity.

The platform's collaborative features, powered by smartphone-based touch interfaces and networking capabilities, enable simultaneous multi-user control, fostering engagement and creativity in group environments. Rigorous usability evaluations and performance testing confirm the system's effectiveness in reducing idle time, enhancing workflow efficiency, and improving the overall user experience.

By blending traditional storytelling methods with modern technology, this work contributes to preserving cultural heritage, while making shadow theater accessible to a wider audience through innovative digital tools.

# Περίληψη

Η παρούσα διπλωματική εργασία εισάγει σημαντικές βελτιώσεις και επεκτάσεις στην ψηφιακή πλατφόρμα αφήγησης eShadow [8] ενσωματώνοντας ροές εργασίας δημιουργίας ψηφιακών μαριονετών με υποβοήθηση τεχνητής νοημοσύνης και λειτουργίες συνεργασίας σε πραγματικό χρόνο. Αυτές οι βελτιώσεις αντιμετωπίζουν περιορισμούς στην προσβασιμότητα, χρηστικότητα και διαδραστικότητα πολλών χρηστών, διευρύνοντας την εφαρμοσιμότητα της πλατφόρμας σε εκπαιδευτικά και καλλιτεχνικά περιβάλλοντα.

Αξιοποιώντας προηγμένες τεχνικές μηχανικής μάθησης και υπολογιστικής όρασης, το προτεινόμενο σύστημα μετατρέπει διάφορες εισόδους—από πραγματικά αντικείμενα έως χειροποίητα σχέδια—σε ψηφιακές μαριονέτες, ξεπερνώντας περιορισμούς προηγούμενων προσεγγίσεων. Η λύση ενσωματώνει επεξεργασία εικόνας μέσω κάμερας υψηλής ανάλυσης, αυτόματη αφαίρεση φόντου και παραγωγική τεχνητή νοημοσύνη για τη δημιουργία αισθητικά συνεκτικών μαριονετών με ελάχιστη παρέμβαση από τον χρήστη. Ένα εξειδικευμένο μοντέλο LoRA (Low-Rank Adaptation) χρησιμοποιείται για την αναπαραγωγή του παραδοσιακού στυλ τέχνης του θεάτρου σκιών, διασφαλίζοντας την αυθεντικότητα του ύφους.

Οι συνεργατικές δυνατότητες της πλατφόρμας, που βασίζονται σε διεπαφές αφής μέσω smartphone και δικτυακές λειτουργίες, επιτρέπουν τον ταυτόχρονο έλεγχο αντικειμένων από πολλούς χρήστες, ενισχύοντας τη διάδραση και τη δημιουργικότητα σε ομαδικά περιβάλλοντα. Εκτενείς αξιολογήσεις χρηστικότητας και δοκιμές απόδοσης επιβεβαιώνουν την αποτελεσματικότητα του συστήματος στη μείωση του χρόνου αναμονής, τη βελτίωση της αποδοτικότητας των ροών εργασίας και την ενίσχυση της συνολικής εμπειρίας των χρηστών.

Συνδυάζοντας τις παραδοσιακές μεθόδους αφήγησης με τη σύγχρονη τεχνολογία, η παρούσα διπλωματική εργασία συμβάλλει στη διατήρηση της πολιτιστικής κληρονομιάς, καθιστώντας το θέατρο σκιών πιο προσιτό σε ένα ευρύτερο κοινό μέσω καινοτόμων ψηφιακών εργαλείων.

# Acknowledgments

I would like to extend my deepest gratitude to Dr. Nektarios Moumoutzis, Laboratory Teaching Staff of the School of Electrical and Computer Engineering (ECE) at TUC. Nektarios guided my work from its inception to completion, offering invaluable feedback throughout the design, development, and testing phases. His insights were instrumental in shaping the application, and he played a crucial role in creating the appropriate environments for testing. I cannot thank him enough for his unwavering support and encouragement throughout this journey.

# Contents

# 1. Introduction

This chapter provides a brief introduction to the somewhat forgotten art of shadow theater, giving some information afterwards on the effort of creating a digital counterpart and advocating as to why this can be an endeavour beneficial to society, especially education. Finally some specifics will be given regarding the eShadow platform and the forthcoming expansion named as eShadow LCE (Live Collaboration Edition).

## 1.1 Introduction to Traditional Shadow Theater

Traditional shadow theater, an ancient form of storytelling, utilizes flat, articulated figures to cast shadows on a translucent screen, creating dynamic narratives through the interplay of light and shadow. This art form has deep-rooted traditions in several countries, notably China, Indonesia, Turkey, and Greece.

In China, shadow puppetry, known as ”pi ying,” 皮影 literally meaning ”leather shadow” dates back to the Han Dynasty (206 BCE –220 CE). It was used to convey folklore, historical events, and moral lessons, often accompanied by music and singing. The puppets, crafted from leather or paper, are manipulated to create the illusion of moving images on a translucent cloth screen illuminated from behind. [58]

Indonesia’s ”wayang kulit” is a renowned form of shadow theater, particularly in Java and Bali. Performances often depict stories from the Ramayana and Mahabharata epics, serving both entertainment and educational purposes by imparting moral and philosophical lessons. The puppets are made from leather and are intricately carved and painted. [61]

In Turkey, ”Karagöz” shadow theater emerged during the Ottoman Empire, featuring humorous and satirical narratives that reflect social and political themes. The performances often involve two main characters, Karagöz and Hacivat, engaging in witty dialogues that entertain and provoke thought. [60]

Greek shadow theater, known as ”Karagiozis,” became prominent during the Ottoman occupation of Greece. The central character, Karagiozis, is depicted as a cunning and impoverished Greek who frequently outwits authority figures. These performances fulfilled several important roles in society. By dramatizing historical events and folklore, they provided audiences with valuable insights into Greek history and cultural heritage. At the same time, the humorous and satirical nature of the stories offered entertainment, making them

widely enjoyed by both children and adults. Additionally, the narratives often served as a form of social commentary, critiquing societal norms and political situations, which encouraged reflection and discussion among viewers.

The evolution of Karagiozis in Greece reflects the country's socio-political changes. Initially influenced by Turkish shadow theater, it gradually incorporated Greek elements, especially after the country's independence in the 19th century. The character of Karagiozis became a symbol of the Greek spirit, embodying resilience and wit. [64]

In contemporary Greece, with the wide adoption of TV's and the rise of the internet, traditional shadow theater has seen a steep decline. One of the last large-scale efforts to push shadow theater into the mainstream was with the karagiozis tv show by Mr. Evgenios Spatharis, which ran from 1966 up to 1992 to overwhelming public acclaim. However, it was inevitably taken over by more modern shows catering better to the interests of newer generations. Nowadays, Greek shadow theater is limited to sporadic performances by local artists. Mr. Spatharis's figures are in prominent display in institutions like the Spathario Museum of Shadow Theater in Athens [57], ensuring that this cultural heritage remains accessible to future generations.

## 1.2 Application of Traditional-Digital Shadow Theater in Education

Shadow theater and theater in general can have a huge contribution potential in education. In its passive form (viewing), a theatrical play can be educative and thought-provocative; however, the form with the biggest education potential is the participatory form. Through a theater play, students can develop their creativity, improve their speaking skills, and learn to interact with others in a constructive way. Shadow theater here can add an extra layer of creativity as it can introduce forms and objects non-replicable using the human body. Additionally, in cases where the students derive from a country with a rich tradition in shadow theater, giving shadow theater a place in the classroom can help preserve this currently disappearing art and cultivate a sense of appreciation for the contributions of shadow theater in modern culture.

One of the biggest obstacles in introducing traditional shadow theater in classrooms is probably the installation and operation overhead of the shadow theater stage, as it can prove to be quite a costly endeavor requiring, in many cases, artistic and technical expertise, most schools simply cannot afford. However, the recent rise and democratization of modern technology has given rise to new, cheaper, and easier ways to replicate shadow theater in classrooms in ways that preserve the core educational benefits it offers. Therefore, there have been a handful of notable endeavors at the international level of taking advantage of those digital technologies to create experiences that can benefit education and popularize shadow

theater once again.

## 1.3 The eShadow Platform and its Limitations

eShadow is a digital storytelling platform inspired by traditional Greek shadow theater, designed to blend cultural heritage with modern technology. It allows users to create, record, and share interactive digital performances using customizable digital puppets, sceneries, and scripts [37]. The platform emphasizes creativity and engagement, providing an accessible way to experiment with storytelling techniques while preserving the artistic traditions of shadow theater. eShadow has been used in educational settings to enhance learning, foster collaboration, and inspire creativity, making it a versatile tool for both formal and informal environments. Its adaptability and user-friendly interface make it suitable for a wide range of audiences, from students to cultural enthusiasts [8][38].

Despite its innovative features, eShadow has limitations that constrain its usability in collaborative and dynamic settings. One major limitation is the lack of real-time collaborative functionality, which prevents multiple users from simultaneously controlling puppets or contributing to a shared scene over a network. This absence of collaborative tools makes the platform less effective in group environments, particularly in educational or creative workshops where teamwork and interaction are key components.

Another significant challenge lies in the complexity of creating and adding new puppets and sceneries. The process requires technical skills, such as isolating, cropping, and assembling puppet parts, as well as generating detailed JSON files to configure their structure. This technical barrier can discourage non-expert users, such as educators or students, from fully utilizing the platform's potential, limiting its accessibility and hindering the creation of personalized content.

## 1.4 eShadow LCE (Live Collaboration Edition)

The eShadow LCE (Live Collaboration Edition) system aims to improve upon the original eShadow in 2 main aspects: puppet creation and puppet manipulation in a local collaborative setting, while keeping the functionality as tightly integrated, responsive, and user-friendly as possible.

The main idea consists of taking advantage of readily available smartphone capabilities such as a high-resolution camera as well as various input and communication interfaces integrated into the devices to expand upon the capabilities the platform.

The puppet creation part is to be able to work with both drawn parts as well as real life objects such as trees, statues and be even able to transform a person into a puppet in a fast but effective way with minimal to no limitations as to what the source is. This goal is to be achieved by using powerfull machine learning and computer vision algorithms running

localy on the host device, as well as a small pinch of diffusion based generative AI that will be able to give a more artistic feel to object or people captured.

The collaborative puppet manipulation aspect aims to allow multiple users to move their puppets simultaneously in real time using their smartphone's touch screen as an input.

More concrete details are to be elaborated on on a later chapter, however the system architecture on its most basic form consists of a computer running the updated eShadow platform acting as a local server connecting over the local network to various mobile devices and exchange information.

## 1.5 Thesis Outline

The thesis is structured to systematically explore the eShadow LCE system, highlighting its potential to integrate modern technology into the traditional art of shadow theater. The document begins with an Introduction, providing an overview of shadow theater's cultural significance and its modern-day educational applications, followed by a detailed look into the current capabilities and limitations of the eShadow platform.

The Research Overview delves into related works, exploring how emerging technologies have been utilized in digital storytelling and their implications for educational and cultural applications. This section also introduces the main frameworks utilized to develop eShadow LCE, as well as key technologies making it's implementation possible.

Next, the Requirements and Use Cases define the functional needs of the system and describe practical scenarios featuring user personas like students and teachers, illustrating how the system is designed to meet their unique requirements.

The System Design and Architectural Overview outlines the technical framework of the expanded platform, emphasizing the integration of advanced features like puppet creation and multi-user collaboration.

The Implementation chapter provides a detailed description of key modules, the technological choices made, and the innovative approaches taken to overcome challenges. This is followed by the Evaluation and Testing chapter, where the system's performance and user feedback are analyzed to assess its effectiveness and identify areas for improvement.

Finally, the Discussion addresses the results in the context of the project's objectives, compares the proposed solution with existing platforms, and explores any limitations. The thesis concludes with a summary of contributions and a roadmap for future work, outlining potential research directions to further enhance the platform's capabilities.

# 2. Research Overview

This chapter is divided into three main sections. The first section discusses existing work in digital storytelling that inspired this project. The second section explains the frameworks used in the development of eShadow LCE, as well as the reasons for choosing them instead of some other competing framework. The third section gives a brief overview of the key technologies and techniques critical to make the development of eShadow LCE possible.

## 2.1 Digital Storytelling Platforms

There have been a few extremely innovative projects that have provided inspiration and guidance during the design and development of the eShadow LCE, some of which i present below.

### 2.1.1 Past eShadow Components And Their Shortcomings

#### 2.1.1.1 eShadow Editor

The eShadow Editor [35] is a desktop application designed to support digital storytelling inspired by traditional Greek shadow theater. It enables users to create and edit digital 2D figures and scenes using vector graphics (SVG) for figures and raster graphics for scenes. The application provides a wide range of pre-designed figure components that users can combine, modify (e.g., color or outline), and customize to create unique characters. Additionally, it allows scene creation by importing images or using integrated tools like Microsoft Paint. Developed in Java with an intuitive graphical user interface, the eShadow Editor integrates traditional shadow theater aesthetics while offering flexibility and ease of use. This tool complements the eShadow platform by easing the arduous process of creating puppets and sceneries from scratch. The primary shortcomings of the eShadow Editor lie in its limited integration with the overall platform and the restricted range of puppet creation options. Users are confined to assembling puppets from a predefined set of components, which can feel creatively restrictive for those seeking more customization or originality. Furthermore, the process of assembling and coloring puppets on the desktop application can be tedious and time-consuming, potentially hindering efficiency and user engagement.

#### 2.1.1.2 ePuppet

The ePuppet [40] application is a mobile solution that bridges traditional and digital shadow theater by enabling users to digitize, customize, and import custom figures and scenery into the eShadow platform. Designed for educators and students, it allows the transformation of analog puppets into digital figures through a mobile device's camera. Users can create figures using predefined templates or opt for free-form creation by segmenting parts and defining joints. Built as a hybrid app using the Ionic Framework and Angular, ePuppet simplifies the figure creation process with an intuitive interface, efficient image processing, and storage on the device or cloud. The application supports both structured and creative workflows, ensuring adaptability for diverse user needs Main shortcoming include that the app can only work on drawings with separated parts on while assuming optimal lighting and monochromatic backgrounds during image capture. Additionally while the application provides cloud-upload functionality via Google Drive, it lacks direct integration with the platform therefore not completely solving the problem.

### 2.1.2 Inspiring Work on Digital Storytelling

#### 2.1.2.1 ShadowMaker

One fairly recent work using the recent diffusion technologies comes from China and is called ShadowMaker [65]. It uses diffusion-based prompt generation that fills pre-existing molds of traditional Chinese puppets, serving as a great source of inspiration for the work presented in this paper. In this thesis, diffusion technologies are utilized to create stylized but faithful recreations of the input image. The main difference is that the combination of models utilized in this thesis are prompt-less by design to reduce the overall complexity on the users end.

#### 2.1.2.2 eHeritage

Another notable work called 'eHeritage of Shadow Puppetry: Creation and Manipulation' [31] features the transformation of human faces into traditional Chinese puppet characters, creating the impression that the individual in the photo has been seamlessly turned into a puppet. This method integrates a personalized puppet creation module with an innovative manipulation framework. The manipulation is rooted in authentic motion sequences derived directly from traditional Chinese theater, preserving its unique stylistic charm while enabling dynamic and lifelike puppet animations. [31]

### 2.1.3 Other Creative Uses of Technology in Digital Theater

There have also been a great deal of interesting ideas in introducing novel ways to interact with the computer that have great potential in enriching the theatrical and storytelling expe-

rience.

One work is about utilizing Leap Motion technology [32] to capture intricate finger gestures, allowing users to control digital puppets in a manner that mimics traditional puppet manipulation. By leveraging inverse kinematics and Unity, users can seamlessly direct the head, arms, and legs of the puppets within interactive 3D environments, bringing a new dimension to the art form [6].

In addition, related work has explored the integration of digital puppetry with formal and informal learning settings, as demonstrated in the Caravan Next project [39]. This project utilized a combination of specialized software and Leap Motion controllers to enable real-time interaction with digital puppets, enhancing both cultural engagement and educational experiences. For example, students in primary schools used Leap Motion devices to animate digital puppets inspired by traditional myths, fostering creativity and teamwork. The accompanying software allowed users to create animated stories by combining scanned drawings, real-time gesture-based puppet control, and interactive storytelling.

Another intriguing application employs Kinect depth sensors [62] to capture full-body movements, enabling audience members to control virtual puppets through their physical actions. A prime example is the "Reconstruction" project, where participant's motions are mapped onto characters inspired by Chinese shadow puppetry. This innovative system reimagines traditional puppetry by allowing users to interact directly with the performance through their gestures, creating a dynamic and immersive experience [29].

A project hosted on the Baidu AI Studio website endeavors to safeguard the fading tradition of shadow puppetry, highlighting its allure and historical significance. Through the innovative integration of Chinese puppet reconstruction and mapping techniques onto the human body, the project utilizes PaddleHub's body keypoint detection library. This integration enables the transformation of human gestures into fluid puppet movements, effectively revitalizing shadow puppetry through the application of AI technology. [3]

Adding a collaborative twist to digital puppetry, the CoPuppet framework [4] distributes control of puppet components among multiple users. Participants can steer different parts of a virtual puppet through gestures, webcams, or voice input, fostering a sense of shared performance. This approach seamlessly integrates traditional and digital puppetry techniques, demonstrated in adaptations of wayang shadow play, where performers collaboratively craft live, improvised storytelling. The creation of a digital interacting platform to create theater plays using traditional Greek Shadow Theater figures [37][36][8], platform to incorporate plays in Turkish theater [22].

## 2.2 Frameworks and Technologies Used

The development of the eShadow LCE system involved integrating several frameworks and technologies, each contributing essential functionalities. This section outlines the key frame-

works and technologies used, their roles in the project, and the rationale behind their selection.

### 2.2.1 Flutter

Flutter [21], a UI toolkit developed by Google, was used to build the mobile application, *eShadow Companion*, which serves as the user interface and a client for communication with other components of the system. Flutter's ability to create cross-platform applications from a single codebase ensures that the application can run on both Android and iOS devices, enhancing accessibility. Flutter uses Dart [20] as its primary coding language.

Flutter simplifies the development of visually appealing and responsive applications through its widget-based approach, providing a rich library of pre-built and customizable widgets. This approach allows for a consistent look and feel across platforms and ensures that the application adapts to the native style of the operating system.


Figure 2.1: Flutter Logo.

The extensive collection of prebuilt libraries and plugins in Flutter, along with strong community support, simplifies tasks such as network communication, API integration, image manipulation, QR code scanning, camera configuration, authentication, and real-time database integration. This reduces development time and effort.

Flutter was chosen over alternatives like React Native and native development (Java/Kotlin or Swift) due to its superior performance, especially in applications requiring high frame rates and smooth animations. Flutter's architecture eliminates the need for a JavaScript bridge (used in React Native), reducing latency and improving responsiveness. In this thesis Flutter is use to build the mobile app that will act as a bridge between the user and eShadow LCE's functionality.

### 2.2.2 Python

Python [49] was selected as the primary programming language for handling complex and resource-demanding functionalities like image processing, running AI models, and server-side operations. Python's flexibility, simplicity, and extensive library ecosystem made it suitable for implementing advanced functionalities such as machine learning workflows and computer vision tasks.

Key libraries used include Flask for creating RESTful APIs, Rembg for background removal, and ONNX Runtime for efficient execution of AI models. Python's versatility allowed for seamless communication between the mobile app and the Unity-based platform.


Figure 2.2: PyTorch Logo.

Deployment was simplified using PyInstaller, packaging the Python server into standalone executables, eliminating the need for users to install Python or manage dependencies. This approach ensures compatibility across different systems, which is crucial in educational environments.

Python was preferred over alternatives like Node.js and Java due to its superior performance in AI and machine learning tasks and its less verbose syntax.

### 2.2.3 PyTorch

PyTorch [15] was chosen as the deep learning framework for implementing resource-intensive tasks, more specifically, in this thesis PyTorch is used to combine multiple generative image AI models to facilitate for prompt-less image style transform. Its dynamic computation graph provides flexibility for experimentation and debugging, making it suitable for both research and production environments. PyTorch's support for GPU acceleration via CUDA enabled efficient execution of computationally demanding processes like image style transformation and segmentation.

Features such as built-in utilities for data loading, model optimization, and distributed computing facilitated the integration of advanced AI models in this thesis, including fine-tuning Stable Diffusion using LoRA as well as combining and running multiple diffusion models in an efficient manner.

Alternatives like TensorFlow and Keras were considered but were less suitable due to reasons like less intuitive debugging and lack of low-level flexibility required for custom implementations.

### 2.2.4 Unity

Unity [59], a leading game development framework, was used to expand the original eShadow platform. Since the base platform was already built in Unity, it was decided to use the same environment to add new features and establish communication with the Python server for the scope of this thesis.


Figure 2.3: Unity Logo.

Unity's flexibility and powerful rendering capabilities made it ideal for implementing features like puppet manipulation and animation. Interfaces were created to allow users to upload their puppets and interact with them dynamically. Unity also managed the collaborative aspect of the platform, enabling multiple users to manipulate puppets simultaneously.

### 2.2.5 LoRA

Low-Rank Adaptation (LoRA) [26] is a technique designed to fine-tune large-scale AI models for specific tasks without retraining the entire base model. It adapts a small subset of the model's parameters, reducing computational and memory requirements.

In this thesis, LoRA was used to fine-tune a Stable Diffusion model to replicate the art style of traditional shadow theater. A training dataset of approximately 12 images of traditional shadow theater figures was used, and the process was completed in about 25 minutes on an NVIDIA RTX 4060 GPU. The resulting LoRA model generates puppet images with a consistent traditional aesthetic, enhancing the platform's ability to transform user-generated images into stylized shadow theater figures.

### 2.2.6 Stable Diffusion

Stable Diffusion [51] is a generative AI model used for image generation and manipulation. It employs a diffusion model that iteratively refines an image from random noise by learning a series of denoising steps.

In this thesis, a fine-tuned Stable Diffusion model was used to transform user-provided images into the traditional shadow theater art style. By training the model on a curated dataset of shadow theater figures, it captures the intricate patterns and textures typical of this art form, ensuring consistency across generated images.

### 2.2.7 ControlNet

ControlNet [68] is an extension of Stable Diffusion that enhances control over the image generation process by incorporating auxiliary conditions such as sketches, poses, or edge maps. This allows users to influence the composition and structure of the generated images more precisely.

In this thesis, ControlNet was used to refine the art style transformation process by utilizing edge maps generated from user-provided images. These edge maps serve as structural guides, ensuring that the stylized output retains the desired shape and details of the input.

### 2.2.8 AI-Assisted Background Removal (Rembg Library)

The background removal functionality in the eShadow platform is powered by the Rembg library [18], which uses a pre-trained deep learning model based on the U-Net architecture.

U-Net [50] is effective in image segmentation tasks due to its ability to capture both high-level and fine-grained features.

In this thesis, this automated background removal is critical for creating clean digital puppets. By clearing the background, we can obtain a clean image containing only the object of interest, to which we can later apply desired operations to turn it into either a puppet or a scenery.

Multiple pre-trained U-Nets were tested in terms of speed and effectiveness, and the one chosen in the end is the unetp.onnx model. The aforementioned model, despite having a size of only 4MB, its results were utterly similar to the standard unet.onnx model, despite having less than 20 times the size. Additionally, the processing speed on the unetp was orders of magnitude faster than most other models tested, making it able to run fast enough to be hosted on a mid to low-range laptop computer without any specialized neural network compute units.

### 2.2.9 Communication Technologies and Protocols Used in the Application

The server in this application is designed to run locally on the same device that operates the eShadow platform, eliminating the need for a static external server. This design leverages the local network to facilitate communication with external devices. The decision to use a local server is driven by several key advantages.

Maintaining an external server can be costly, not only in terms of hosting but also in ensuring robust security. A hacked external server could lead to resource exploitation, resulting in financial losses. In contrast, a local server eliminates these concerns by operating within a restricted, secure network environment. Additionally, local servers offer near-zero latency, enabling faster and more responsive communication between devices. Another significant benefit is that the system can function without requiring an external internet connection, making it highly reliable in scenarios where internet access is limited or unavailable.

To enable communication between the various components of the app, several network technologies and protocols are utilized, with the most fundamental ones outlined in this section.

#### 2.2.9.1 HTTP Protocol

HTTP (HyperText Transfer Protocol) is a widely used protocol for communication over the web. It is designed for transferring data between a client (such as a web browser or mobile app) and a server. HTTP operates over a connection-oriented protocol (usually TCP) to ensure reliable data delivery, making it ideal for applications requiring structured data exchange. In this thesis, HTTP is used to transport reliably data such as images.

#### 2.2.9.2 UDP Protocol

The User Datagram Protocol (UDP) is a lightweight, connectionless protocol designed for transmitting data without the need to establish a dedicated connection. Unlike HTTP, which prioritizes reliability, UDP focuses on speed and low latency, making it ideal for scenarios such as real-time data streaming or broadcasting messages across a network.

In the context of this thesis, UDP is employed to transmit low-latency messages that control the figures displayed on the screen. Given that the application operates within a local network, the need for a more complex or advanced protocol is unnecessary, as packet loss in this controlled environment is negligible. This design choice ensures efficient communication with minimal overhead, perfectly aligning with the application's real-time requirements.

#### 2.2.9.3 Flask

Flask is a lightweight, Python-based web framework used for developing web applications and APIs. It is well-suited for local network applications, as it provides tools to quickly build servers capable of handling HTTP requests. In this thesis, Flask serves as the backbone for the server, enabling seamless interaction with client devices on the local network.

#### 2.2.9.4 QR Codes

QR codes (Quick Response codes) are two-dimensional barcodes that can encode a variety of information, such as text, URLs, or other data. Unlike traditional barcodes, which store information in a single horizontal line, QR codes store data in both vertical and horizontal dimensions, allowing them to hold significantly more information.

A typical QR code consists of black squares arranged on a white background, forming a matrix. These codes are designed to be scanned and decoded using devices like smartphones or QR scanners. They are widely used due to their speed and versatility, enabling instant access to data, such as opening links, connecting to Wi-Fi networks, or initiating app downloads.

In this thesis, QR codes play a vital role in establishing connectivity within the local network. Since the application operates in a local network environment, the server's IP address and port are dynamically assigned by the router, which can vary with each session. To simplify the connection process, the server generates a QR code upon successful startup.

This QR code, displayed in the bottom left corner of the interface, encodes the server's current IP address and port. Users can scan this QR code using the mobile application, which performs a quick handshake to seamlessly establish a connection to the server. This approach eliminates the need for manual entry of connection details, ensuring a faster, error-free setup while maintaining the ease of use.

### 2.2.10 PyInstaller Packaging Library.

In order for the Python server application to be able to be utilized by people with little technical knowledge, the need to find a way to package the program along with its dependencies and the background removal model into a portable and easy to use format arose. PyInstaller library [19] was deemed for the scope of this thesis the most suitable, as it can package all dependencies into a single executable featuring multiple configuration options.

### 2.2.11 ONNX Runtime

In this thesis ONNX Runtime is used for running pre-trained machine learning models efficiently on the server, more specifically the background removal model. It supports various hardware acceleration techniques, including GPU and CPU optimizations, ensuring that computationally intensive tasks like image segmentation and style transfer can be performed with minimal latency.

### 2.2.12 Pillow (PIL)

Pillow [48], a fork of the Python Imaging Library (PIL), was used for image manipulation tasks such as format conversion, resizing, and preprocessing before advanced operations like segmentation or style transformation. It provides essential functionality to ensure compatibility and optimization of image data across different processing stages. In this thesis Pillow is used to carry out various image processing oprerations such as image resize and image connected component analysis.

### 2.2.13 CUDA Acceleration

In this thesis CUDA [44] acceleration is utilized to harness the powerful computing capabilities of GPUs for handling computationally intensive tasks, more specifically for accelerating the image style transform process. Without CUDA's GPU acceleration, algorithms such as Stable Diffusion, which rely on frameworks like PyTorch, would require an impractically long time to execute. CUDA significantly reduces processing time, making these tasks feasible.

Furthermore, training a LoRA model, such as the one integrated with Stable Diffusion to generate realistic Greek shadow figures, would be nearly impossible on consumer-grade hardware without CUDA. This technology enables efficient training by offloading the computational burden to GPUs, thus making advanced AI-driven generative tasks accessible and practical.

### 2.2.14 Kohya-ss

Kohya-ss [30] is a specialized tool used for fine-tuning diffusion models and creating LoRA adaptations. It offers a streamlined workflow for training models on smaller datasets. In the eShadow project, in this thesis, Kohya-ss was employed to fine-tune the Stable Diffusion model on images of traditional shadow theater, ensuring the resulting LoRA model accurately captured the distinct artistic elements of the style. A GUI for Kohya-ss is also freely available [30].

## 2.3 Summary

The integration of these frameworks and technologies enabled the eShadow platform to provide an interactive and immersive digital shadow theater experience. By leveraging modern tools and AI-driven functionalities, the platform bridges traditional art forms with contemporary digital capabilities, enhancing accessibility and educational potential.

# 3. Functional Specifications, Use Cases and Personas

This chapter defines the functional specifications, use cases, and personas that guided the development of the enhanced eShadow platform. It outlines the key features required to meet the platform's objectives, such as facilitating collaborative puppet creation, enabling real-time manipulation, and integrating AI-driven functionalities for seamless user experiences.

The use cases provide practical examples of how users interact with the system in real-world scenarios, focusing on educational and creative applications. To ensure the platform addresses diverse needs, personas representing typical users, including students, educators, and artists, are introduced. These personas highlight the goals, challenges, and expectations that shaped the platform's design.

Together, these components form a structured framework for understanding the platform's functionality and its alignment with user needs, setting the stage for subsequent chapters on implementation and evaluation.

## 3.1 Functional Specifications

Functional specifications define the core features and actions a system must perform to meet its intended purpose. They describe what the system does, focusing on user needs and system requirements without detailing how the functionality is implemented. By outlining essential behaviors, functional specifications serve as a blueprint for development and evaluation.

The following functional specifications outline the key features and behaviors of the eShadow LCE (Live Collaboration Edition) application. These specifications ensure the system meets user needs and achieves its intended purpose.

1. **Puppet and Scenery Creation:**
    - The application must enable users to create puppets and sceneries from photos.
    - The generated puppets and sceneries must integrate seamlessly with the eShadow platform.
2. **Collaboration and Connectivity:**

- The system must generate a QR code with the server's IP address and port to allow users to establish a connection using the mobile app.
- The application must support simultaneous connections and real-time interactions for multiple users over a local network.

3. **Control and Interaction:**
   - Users must be able to load puppets and scenery into a shared stage and control them via the mobile app using touch gestures.
4. **Aesthetics and Usability:**
   - The app must be easy to understand and pleasant to the eye.

The application must also adhere to established usability principles, as outlined in [34], [42], and [41], which emphasize intuitive, error-tolerant, and user-friendly interfaces. To this end:

1. **Visibility of System Status:** Users must be kept informed about the system's state through real-time indicators, progress bars for computational tasks, and descriptive error messages.
2. **Match Between System and the Real World:** The application must use intuitive workflows and terminology that mirror real-world processes, reducing cognitive load.
3. **User Control and Freedom:** Users must have the freedom to undo, redo, or cancel actions, ensuring they can easily recover from mistakes or modify their workflows.
4. **Consistency and Standards:** The application must maintain a uniform design and terminology, ensuring predictable interactions and reducing user confusion.
5. **Error Prevention:** The system must validate user inputs and provide warnings for potentially problematic actions to minimize errors before they occur.
6. **Recognition Rather Than Recall:** The application must present all necessary options and information visually to reduce the need for users to remember details or actions.
7. **Flexibility and Efficiency of Use:** The application must cater to both novice and experienced users by providing interactive tutorials for beginners and shortcuts for advanced tasks.
8. **Aesthetic and Minimalist Design:** The interface must prioritize essential elements and avoid unnecessary complexity to prevent user overwhelm.

9. **Help Users Recognize, Diagnose, and Recover from Errors:** The system must provide clear and actionable error messages, along with tools for troubleshooting and recovery.

10. **Help and Documentation:** The application must include interactive tutorials, FAQs, and a comprehensive help section accessible at any point in the user journey.

## 3.2 Use Cases

Use cases describe interactions between users and the system, capturing how the system responds to specific user actions. They illustrate workflows and scenarios where the system fulfills its intended functions, highlighting its practical applications. Use cases are crucial for understanding the usability and effectiveness of the system in real-world contexts.

### 3.2.1 Mobile App Use Cases

#### 3.2.1.1 Take Picture

The user wants to add a new picture to their gallery. They navigate to the gallery section of the app and tap the floating button with the camera icon. The app opens the camera, allowing the user to take a photo. Once captured, the picture is automatically saved and displayed in the gallery.

#### 3.2.1.2 Toggle Big Picture View

The user wishes to enlarge an image for better viewing. They tap on the image they want to see in a larger format. The image expands to a full-screen view. Tapping anywhere on the screen again minimizes the image back to its original size.

#### 3.2.1.3 Connect to the Servers

The user needs to connect the app to a server. They tap the QR icon in the upper left corner and proceed to scan the QR code provided to them. The app detects whether the code corresponds to the AI server or the main server. Once connected, the appropriate indication flashes, and new options are made available depending on the server.

#### 3.2.1.4 Create Figure From Drawn Parts

The user wants to create a figure using parts drawn on paper. They navigate to the gallery and long-press the desired image. A drop-down menu appears, and the user selects the *Create Figure* option. They are prompted to choose between creating a figure *from paper* or *something else*. Selecting *from paper*, the user follows a two-step process. First, they crop

the image to remove unnecessary elements and center the subject, then tap *Next*. The app segments and separates the parts automatically. The user arranges the parts, defines joints by long-pressing intersections, and taps *Finish*. The figure is then generated, along with the necessary files to integrate it into the eShadow platform.

#### 3.2.1.5 Create Figure From Object or Person

The user wants to create a figure from an image of an object or person. They navigate to the gallery, long-press the desired image, and select *Create Figure* from the drop-down menu. After choosing *something else*, the user follows a four-step process. They crop the image to remove unnecessary elements and center the subject, then tap *Next*. The app automatically removes the background, allowing the user to clean up any leftovers if necessary. In the third step, the user selects the areas to become figure parts by circling them and tapping *Next*. Finally, the parts are displayed in a scene where the user arranges them, defines joints, and taps *Finish*. The figure and associated files are generated for use in the eShadow platform.

#### 3.2.1.6 Create Scenery

The user wants to create a scenery from an image. They navigate to the gallery, long-press the desired image, and select *Create Scenery* from the drop-down menu. The image is then processed into scenery for use in the app.

#### 3.2.1.7 Load Figure to Scene and Control

The user wishes to load a figure into the scene for control. They navigate to the *Figure Menu*, long-press the desired figure, and select *Load to Stage* from the drop-down menu. The user is taken to the control screen, where they can move the figure by sliding their finger on the screen, change its direction with a two-finger tap, and resize it using the slider below.

#### 3.2.1.8 Load Scenery to Scene and Control

The user wants to load scenery into the scene for control. They navigate to the *Scenery Menu*, long-press the desired scenery, and select *Load to Stage* from the drop-down menu. In the control screen, the user moves the scenery by sliding their finger on the screen and zooms in or out by pinching.

#### 3.2.1.9 Delete Picture, Figure, or Scenery

The user wants to delete an item from the gallery or menus. They long-press the item, select *Delete* from the drop-down menu, and confirm the deletion when prompted. The item is then permanently removed.

#### 3.2.1.10 Change Art Style of an Image

The user wishes to apply an art style change to an image. They navigate to the gallery, long-press the desired image, and select *Change Art Style* from the drop-down menu. If connected to the AI server, the app processes the image and adds the styled version to the gallery as a new item.

#### 3.2.1.11 Upload Puppet to Platform

The user wants to upload a figure to the eShadow platform. They navigate to the *Figure Menu*, long-press the desired figure, and select *Upload to Platform* from the drop-down menu. They are prompted to name the figure before uploading it.

#### 3.2.1.12 Upload Scenery to Platform

The user wants to upload scenery to the eShadow platform. They navigate to the *Scenery Menu*, long-press the desired scenery, and select *Upload to Platform* from the drop-down menu. The scenery is uploaded.

#### 3.2.1.13 Get Hint

The user needs guidance. They press the *Hint* button, which displays context-specific tips depending on the screen they are on. Repeated presses provide different hints.

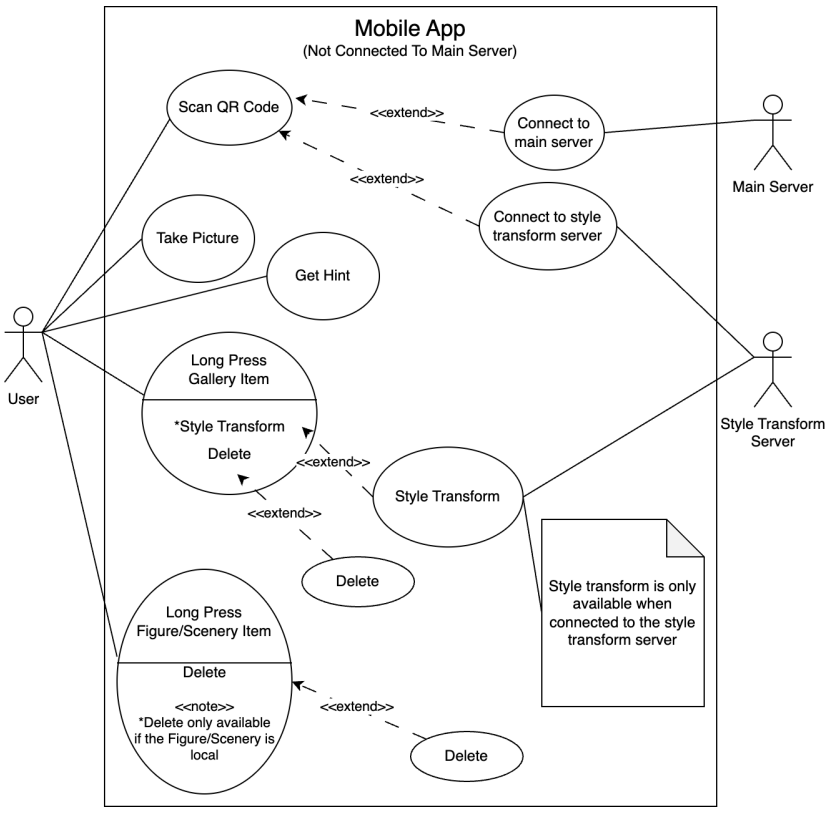



Figure 3.1: Not Connected to Main Server Mobile App Use Case Diagram

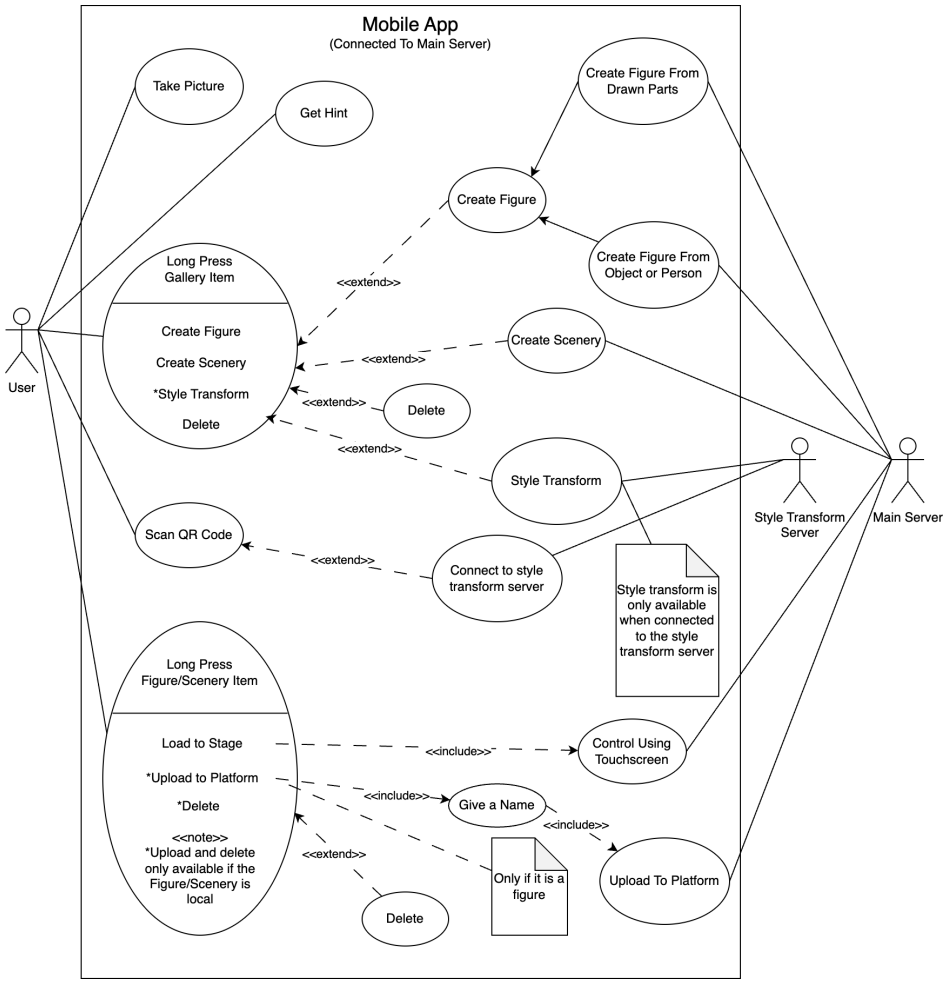



Figure 3.2: Connected to Main Server Mobile App Use Case Diagram

## 3.2.2 Upgraded eShadow Platform Use Cases

### 3.2.2.1 All Existing Functionality

All previously available functionality within the platform remains accessible, allowing users to seamlessly continue their workflow.

### 3.2.2.2 Enable/Disable User Uploads

The user wants to control whether others can upload content to the platform. They navigate to the *Home Menu*, then to *Options*, where they use a toggle switch to enable or disable user uploads.

### 3.2.2.3 Clear Scene

The user needs to reset the scene to its initial state. They press the `R` key on the keyboard, clearing all objects from the scene.

#### 3.2.2.4 Remove Specific Scenery or Figure

The user wishes to remove a specific object from the scene. They hold the `Ctrl` key, move the cursor over the object to highlight it, and click to delete it.

#### 3.2.2.5 Toggle UI

The user wants to hide or display the user interface elements on the screen. They press the `T` key on the keyboard to toggle the visibility of the UI.

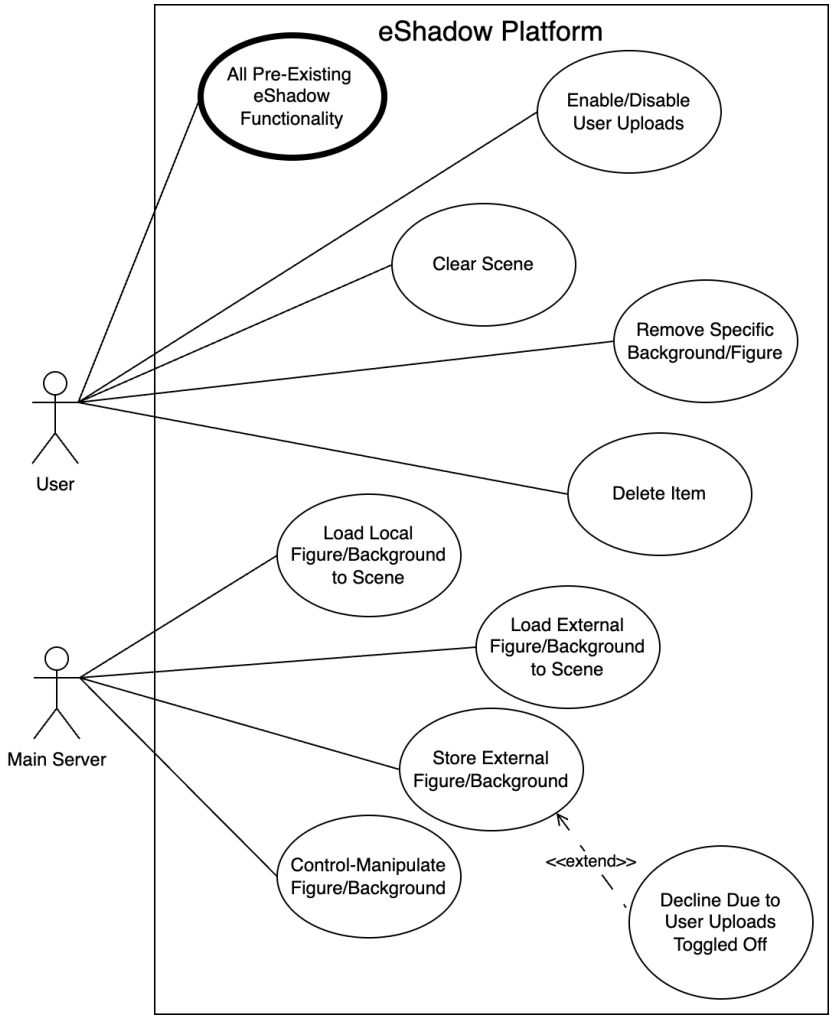



Figure 3.3: eShadow Platform Use Case Diagram

## 3.3 Personas

Personas are archetypal representations of users, designed to reflect their goals, behaviors, and challenges. They help designers and developers understand the diverse needs of the audience and ensure the system is tailored to those needs. Personas are a vital tool in creating user-centered designs that prioritize usability and satisfaction.

### Mary, The Elementary School Teacher Who Likes To Experiment With New Teaching Methods:

**Age:** 31

**Background:** Mary is a passionate and innovative elementary school teacher. She is known for her creative approach to education and her willingness to experiment with new teaching methods to engage her students. Mary believes in the power of hands-on learning experiences and often incorporates art, technology, and other interactive activities into her lessons to make learning more enjoyable and meaningful for her students.

**Goals:** Mary's primary goal is to create a dynamic and inclusive learning environment where every student feels motivated and empowered to explore their interests and talents. She is always on the lookout for innovative tools and resources that can enhance her teaching and provide her students with new opportunities for growth and creativity.

**Behaviours:** Mary has an average understanding of how to use a computers and can accomplish relatively simple tasks such as composing a word file, creating a presentation or installing a piece of software, however she has no insight into coding and has only a superficial understanding of how internet apps work.

**Challenges:** Mary is responsible for setting up the main application along with the projector, troubleshooting any issues, and instructing/supervising the students on how to use the app. She definitely wants to avoid the hassle of interacting with complex installation instructions and prefers the whole process to be easy to understand.

**Needs:** Mary needs a simple way to install and set up the whole system, as well as an easy-to-read short documentation book in case she needs help.

**How the App Addresses Needs:**

- **User-Friendly Installation:** The app is to provide a straightforward installation process with clear, step-by-step instructions that do not require advanced technical skills.
- **Simplified Setup:** The app is to include an intuitive setup wizard that guides Mary through the initial configuration, ensuring that all necessary components are correctly installed and operational.
- **Comprehensive Documentation:** The app is to come with a concise and easily understandable documentation book, offering quick reference guides and troubleshooting tips to help Mary resolve any issues she might encounter.
- **Interactive Tutorials:** To facilitate learning, the app is to feature interactive tutorials and demonstrations that Mary can use to familiarize herself and her students with the app's functionalities.

**Quote:** I only want the best for my students.

## Little Jimmy, The Student

**Age:** 10

**Background:** Jimmy is a 10-year-old student attending a local elementary school. He enjoys participating in various extracurricular activities, but mostly he prefers to stay at home playing with his mother's phone.

**Goals:** Have fun while learning new things.

**Behaviours:** He has a moderate understanding of how to use a mobile device and can accomplish simple tasks such as downloading an app, however might need guidance for more complex one's. Requires information in an easy-to-understand format.

**Challenges:** As a 10-year-old, Jimmy may have limited technical skills and may find it challenging to navigate complex software applications on his own. He may also struggle with focusing for extended periods and may need guidance to stay on track with his creative endeavors.

**Needs:** Jimmy needs an easy to understand intuitive interface without too many options in a streamlined worflow with clear guidance on what to do next, while at the same time providing the option to undo in case something was done wrong.

**How the App Addresses Needs:** The app is (to be) designed with Jimmy's age and skill level in mind, ensuring a user-friendly and intuitive experience tailored to young users. Here's how the app addresses his specific needs:

- **Simple and Intuitive Interface:** The app is to feature a clean, simple interface with large icons and easy-to-understand labels. This design helps Jimmy navigate the app without feeling overwhelmed or confused by too many options.
- **Guided Workflow:** Each step in the app is to be clearly outlined, with instructions provided in a child-friendly manner. This guided approach ensures that Jimmy knows exactly what to do next, reducing the chance of errors and helping him stay on track.
- **Interactive Tutorials:** The app is to include interactive tutorials and baked in hints that walk Jimmy through more complex tasks, such as creating a project or using a new feature.
- **Undo Feature:** Understanding that mistakes are a part of learning, the app is to include a simple undo feature. This allows Jimmy to easily revert any actions if he makes a mistake, fostering a safe environment for exploration and creativity.

**Quote:** I just wanna playyy

## Alexandra, The Artist

**Age:** 27

**Background:** Alexandra is an artist proficient in many art forms such such as painting, cinematography, writing. Alexandra is passionate about art, however, she is even more passionate about using art as a medium to convey emotion and and raise awareness on societal issues.

**Goals:** Alexandra aims to use the platform to incorporate her art in a digital play that conveys emotion, and use the puppet aesthetic as a core feature in her play.

**Behaviours:** Alexandra is highly creative and detail-oriented. She enjoys experimenting with different artistic styles and spends significant time refining her work to achieve a polished final product.

**Challenges:** Alexandra may struggle with overly technical setups and prefers tools that offer intuitive interfaces and minimal configuration.

**Needs:** Alexandra requires a user-friendly platform that enables her to focus on the creative aspects of her work without worrying about technical complexities. **How the App Addresses Needs:**

- **Easiness of Figure Creation:** The app include a comprehensive set of flows that can turn any drawing into a puppet. The app's capabilities are not limited to paper, Alexandra can even include pieces of cloth or other materials as puppet parts.
- **Intuitive Interface:** A streamlined and visually appealing interface ensures Alexandra can navigate the app with ease and focus on her creative process.
- **High-Quality Outputs:** Advanced background removal capabilities included, along with the comprehensive set of creation tools provided allows Alexandra to create high quality figures for her play.

**Quote:** "I am always looking out for new ways to convey emotion."

## 3.4 Task Analysis And Scenarios

Task analysis involves breaking down user activities into sequential steps to understand how specific objectives are achieved. It identifies potential challenges, optimizes workflows, and ensures system efficiency. Scenarios complement task analysis by placing these activities in real-life contexts, demonstrating how the system supports user goals.

### 3.4.1 Task Analysis For Mary

**Task: Set up the System**

**Steps:**

1. Install the eShadow application on her laptop.

2. Connect the laptop to the classroom projector and ensure it is properly configured.
3. Connect eShadow to the main server and display the QR code in the projected scene.
4. Help her students connect their phones to the server.
5. Help her students create their puppets.
6. Organize and record a digital theater performance by having her students perform with their digital puppets.

**Scenario:** Mary is setting up the eShadow system for her class to facilitate a play devised along with her students. She installs the eShadow app on her laptop and connects it to the classroom projector, ensuring it is properly configured. She connects the app to the main server and displays the QR code in the projected scene for her students to scan. She then assists her students in connecting their phones to the server and guides them through the process of creating their digital puppets using the app. Once the puppets are ready, she organizes and records their digital theater performance, where her students use their digital puppets to perform a shadow theater play, creating an engaging and interactive learning experience.

### 3.4.2 Task Analysis For Little Jimmy

**Task: Create a puppet and load it to eShadow**

**Steps:**

1. Open the mobile application on the smartphone.
2. Take a picture of the desired object or drawing using the app's camera function.
3. Use the background removal functionality to isolate the object from its background.
4. Erase any undesired leftovers.
5. Apply art style transformation.
6. Utilize finger gestures to create small gaps and segment the puppet into parts.
7. Assemble the puppet, applying the integrated AI out-paint functionality to complete missing parts wherever needed.
8. Save the puppet within the app.
9. Load the puppet into eShadow for use in a shadow theater play.

**Scenario:** Little Timmy has devised a scenaro for a play along with their teacher in class. He opens the eShadow mobile app on his smartphone and takes a picture of his drawing. Using the app, he removes the background, erases any undesired leftovers, and applies an art style transformation to match the traditional shadow theater aesthetic. He then segments the puppet into parts using finger gestures, assembles the puppet while completing any missing parts with the outpaint feature, and saves it. Finally, he loads the puppet into eShadow, ready to use it for his storytelling activity in class.

### 3.4.3 Task Analysis For Alexandra

**Task: Create a Stylized Puppet Performance**

**Steps:**

1. Sketch or photograph an initial concept for the puppet.
2. Upload the image to the app and use the background removal feature to isolate the puppet.
3. Refine the puppet using the app.
4. Save the puppet and upload it to the Unity platform.
5. Arrange the puppets in a creative digital scene.
6. Record the performance and export it for showcasing.

**Scenario:** Alexandra is preparing a shadow theater performance to rise awareness for a societal issue. She created her puppets segmented in parts and uses the app to create digital puppets.She uploads her hand-crafted designs to the Platform, and arranges them in an elaborate scene. She records the performance adding edits on other software to turn the final video into a piece of art.

# 4. Design Methodology

This chapter presents the methodology and evaluation processes undertaken during the development of the enhanced eShadow platform. The focus is on iterative design, prototyping, and usability testing to ensure that the platform meets its functional requirements and provides a seamless user experience.

The chapter begins with a discussion of the initial prototyping stages, highlighting the evolution from a crude functional prototype to refined user interface designs. The iterative development process, including the use of the Think-Aloud method and trial-and-error techniques, is detailed to emphasize the role of continuous feedback in shaping the platform.

To assess the usability and effectiveness of the platform, heuristic evaluations and structured user testing were conducted. These methods provided critical insights into user interaction patterns, areas of confusion, and opportunities for improvement. The findings from these evaluations are discussed, linking them to the platform's design objectives and the broader goals of fostering collaboration and creativity in shadow theater.

By combining qualitative and quantitative evaluation techniques, this chapter outlines the systematic approach taken to refine the eShadow platform and ensure its alignment with user needs and expectations.

## 4.1 Iterative Development

The iterative development process focused on refining the eShadow platform through repeated testing and improvements. Key enhancements were made to ensure the system was intuitive and efficient, addressing user needs identified during development cycles. Each iteration incorporated design refinements and technical optimizations to align the platform with its usability and functionality goals.

## 4.2 Heuristic Evaluation

To ensure the platform's usability, a heuristic evaluation was conducted based on Nielsen's usability principles. This evaluation focused on identifying potential design issues and improving user interaction. The following heuristics were assessed:

- **Visibility of System Status**: Ensuring that users are informed about the app's status through clear progress indicators, error messages, and tutorial cues.
- **Match Between System and the Real World**: Using intuitive workflows and terminology that align with user expectations.
- **User Control and Freedom**: Incorporating undo/redo options and clear navigation paths.
- **Consistency and Standards**: Maintaining a uniform design language and terminology throughout the platform.
- **Error Prevention and Recovery**: Highlighting potential errors in advance and offering actionable recovery options.
- **Flexibility and Efficiency of Use**: Providing shortcuts for advanced users while ensuring an intuitive experience for novices.

The heuristic evaluation highlighted several areas for improvement, including better visibility for key features like the QR code scanner and clearer feedback mechanisms for completed actions. These insights were integrated into subsequent iterations to enhance usability.

## 4.3 Think-Aloud Evaluation

The Think-Aloud method was employed to gain qualitative insights into user interactions with the eShadow platform. Participants were asked to verbalize their thoughts while performing specific tasks, allowing developers to identify usability issues in real-time. The methodology involved:

1. **Task Design**: Participants completed predefined tasks, such as creating puppets, uploading sceneries, and interacting with the collaborative stage.
2. **Observation**: User interactions were recorded to document pain points, areas of confusion, and intuitive workflows.
3. **Feedback Analysis**: Verbal comments and behavioral cues were analyzed to identify recurring themes.

Key findings included:

- Participants struggled to locate certain features, such as the QR button, indicating the need for more prominent visual cues.
- Suggestions for additional guidance, like contextual hints and step-by-step tutorials, were frequently made.

- Usability ratings varied across participants, reflecting the platform's evolving nature and the importance of addressing specific user needs.

These insights were instrumental in refining the platform's design and improving user satisfaction.

## 4.4 Initial Crude Prototype

The initial crude prototype was created after drafting a rough list of requirements and some basic ideas on how the application is supposed to work. The main goal of the prototype was to test the feasibility of the features and help decide on which to keep on the final design. Creating the prototype was also very helpful in familiarizing with the underlying technologies what would be in turn used to implement the finalized design later on.

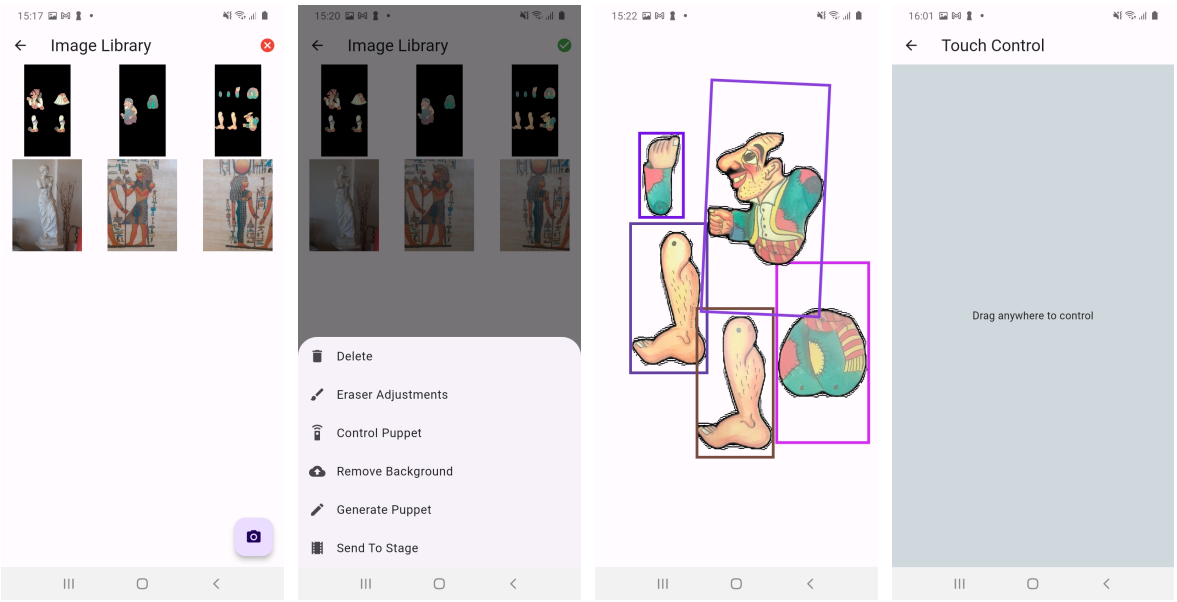



Figure 4.1: Crude Prototype Screens

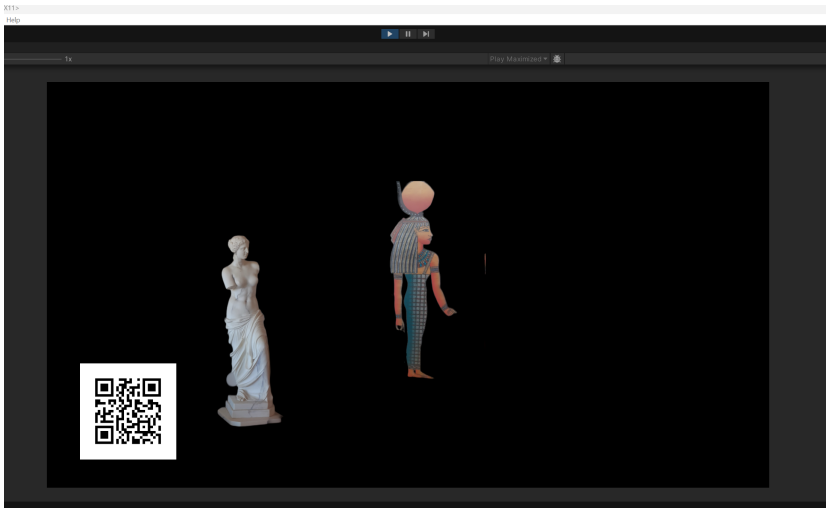


Figure 4.2: Prototype Unity Stage

## 4.5 Figma Prototype

After the initial prototype validated initial proposed features feasibility, a more formal approach was followed by developing an interactive prototype of the application's interface using the Figma [16] platform.

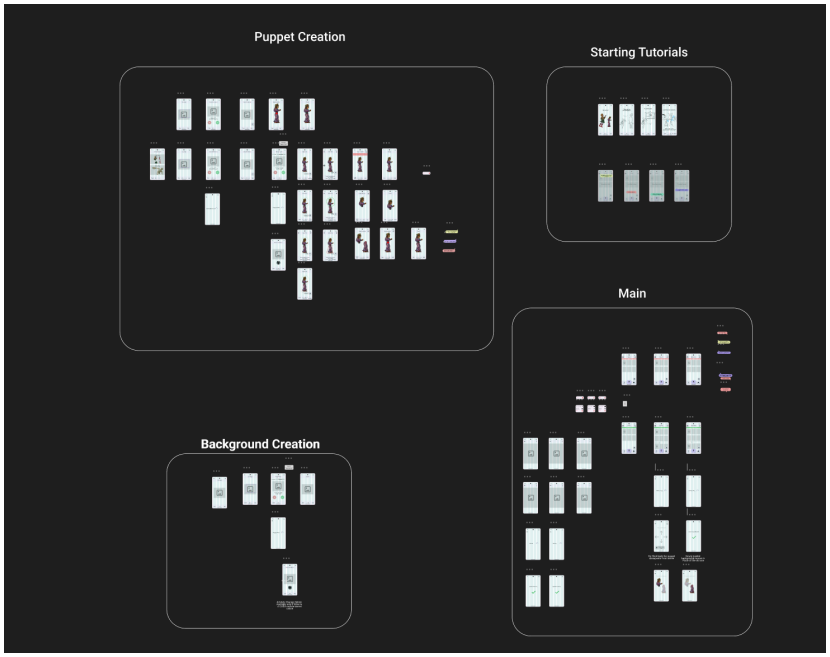



Figure 4.3: High Level Application Screen Overview

Figma provides the ability to create an interactive prototype of an application get a clearer view of the application's design.

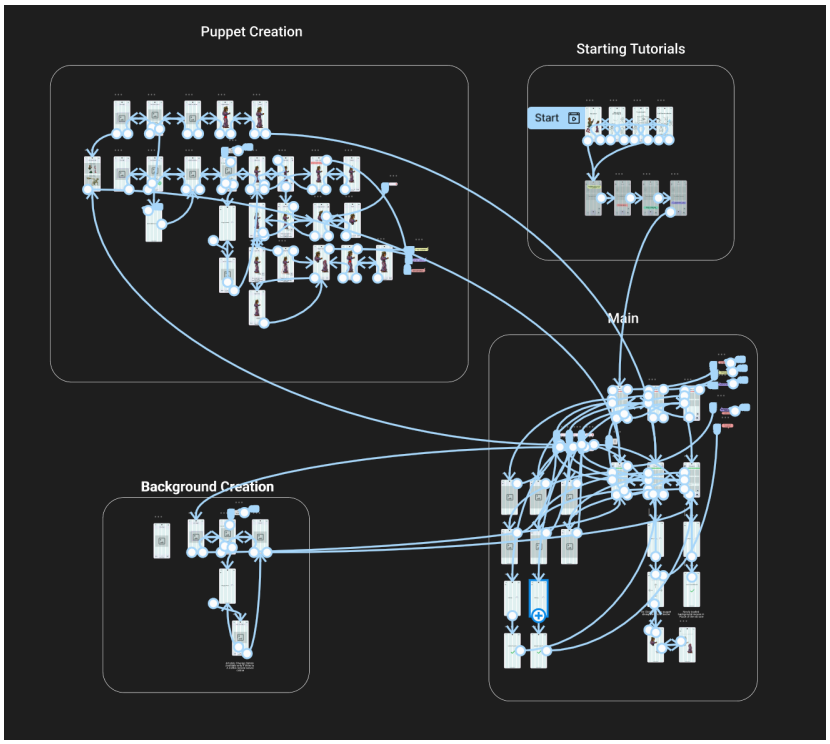



Figure 4.4: Screen Transition Arrows

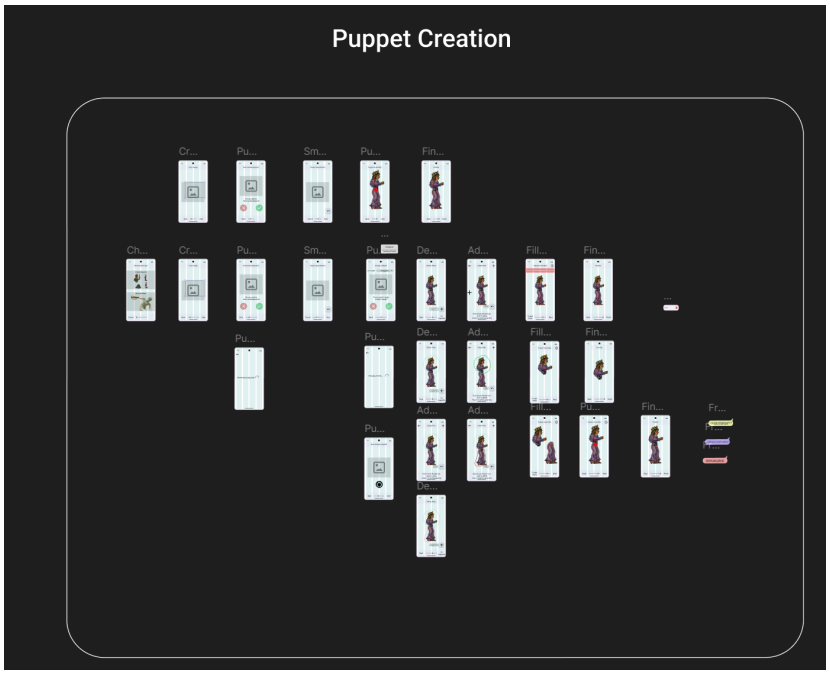



Figure 4.5: Puppet Creation Screens

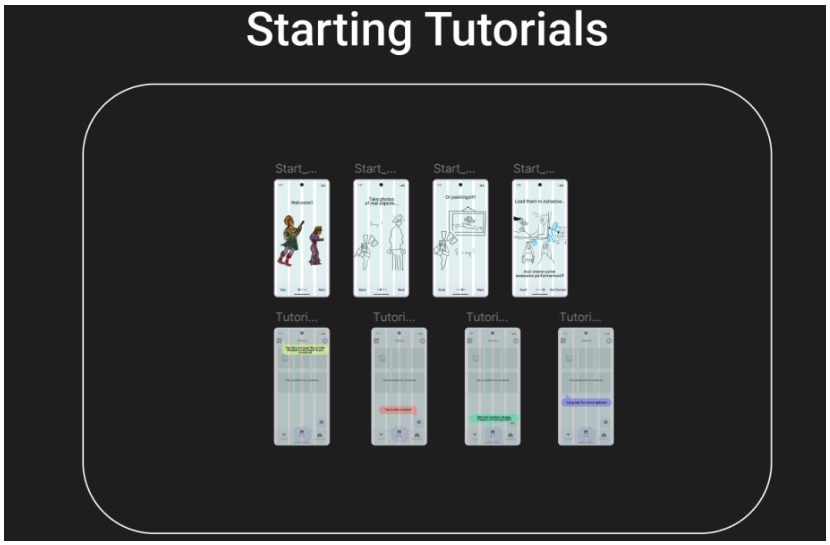



Figure 4.6: Interactive Tutorial Screens

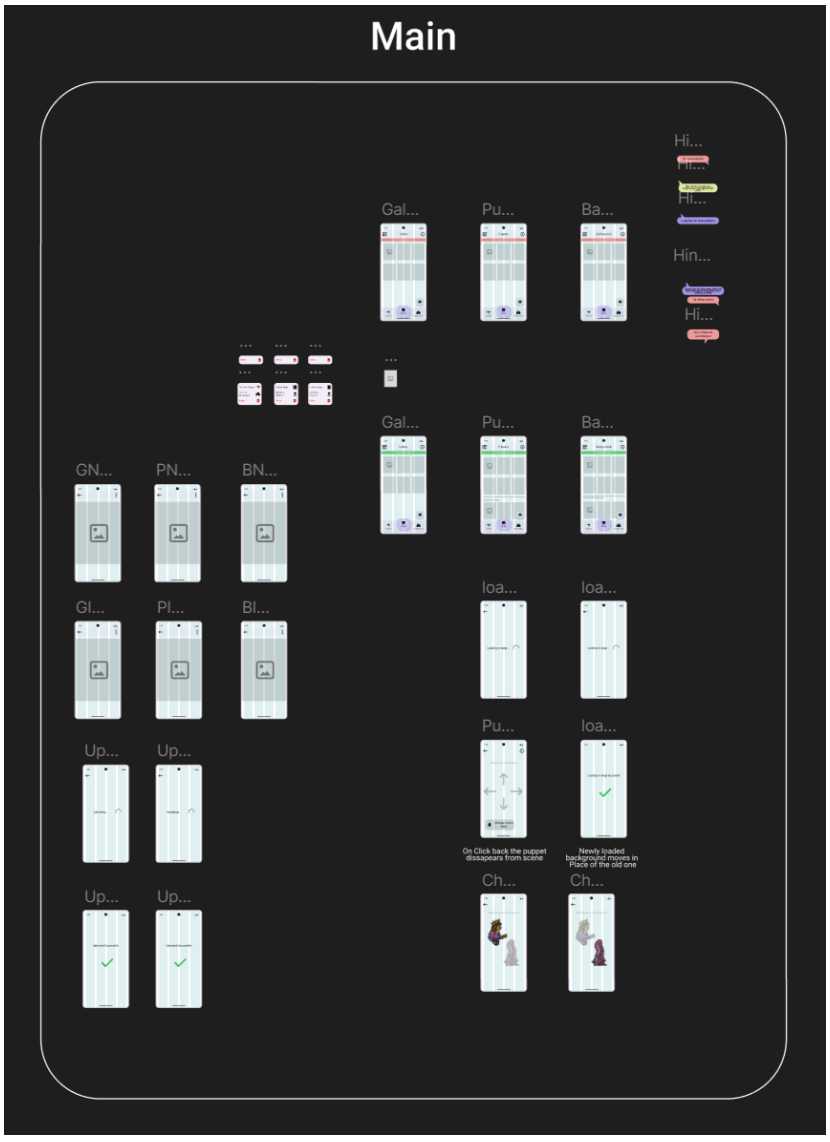



Figure 4.7: Main Screen Interface and Menus

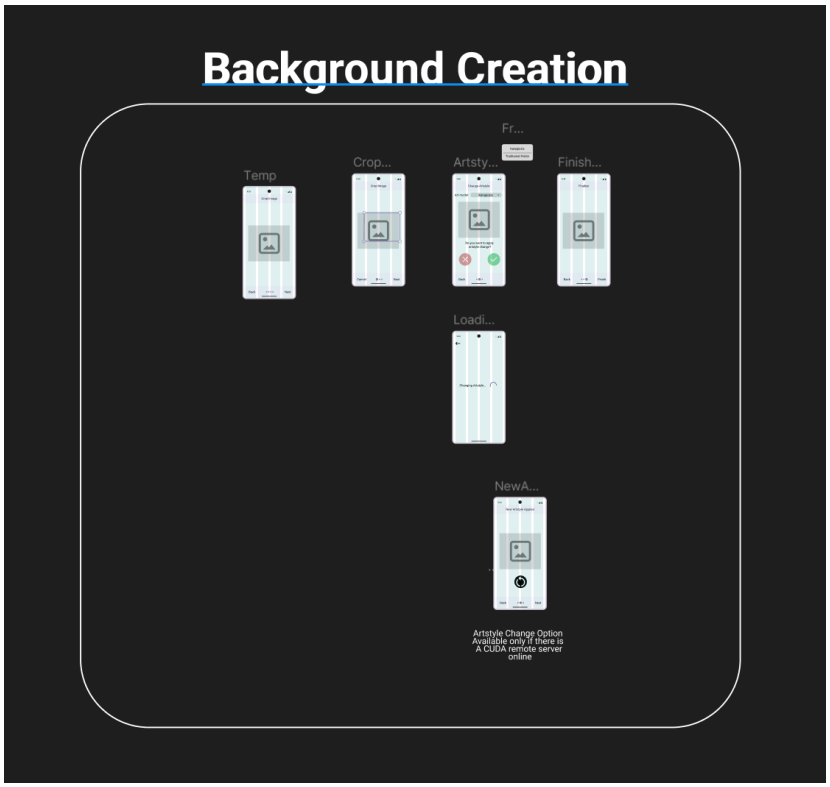
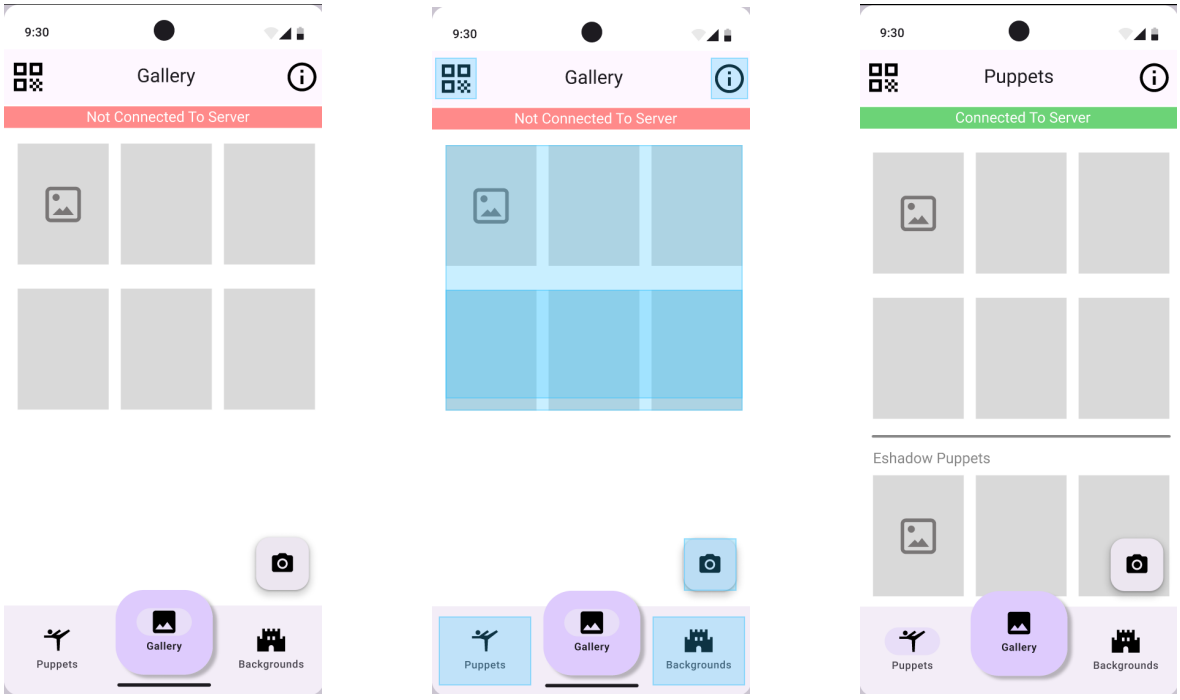
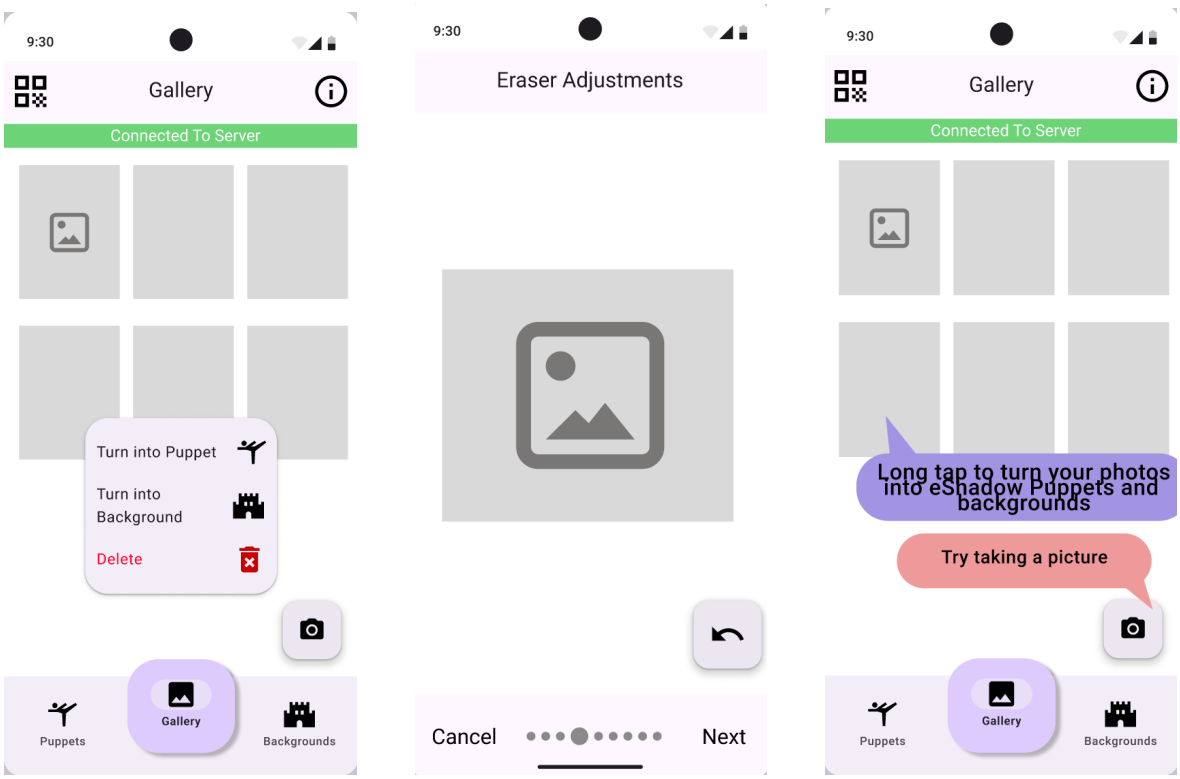



Figure 4.8: Background Creation Screens




(a) Prototype Main Screen (b) Intractable UI (c) Press QR to Connect

Figure 4.9: Example of Prototype Interaction




(a) Tap Image for Menu (b) Figure Creation Process (c) Hints

Figure 4.10: Prototype Sample Functionality

## 4.6 Iterative Development Using Think-Aloud Method

To optimize usability, we employed an iterative development approach that incorporated user feedback at each stage. The think-aloud testing method was used to identify and resolve usability issues. Below are insights gathered from different testers:

#### 4.6.0.1 Retired Navy Officer, 62 years old

- Tutorial completed, but lacked clarity on task completion status.
- Observed a missing indication for tutorial completion details.
- Suggested adding a ”Press anywhere to continue” prompt to tutorial bubbles at the start.
- Recommended graying out the interface to indicate that inputs are disabled until the tutorial finishes.
- Requested guided hints throughout the interface.
- Suggested improving the visibility of the QR button.
- Proposed making clickable areas more apparent.

- Rated usability as **3/10**, citing a need for more detailed instructions.

#### 4.6.0.2 Cooking Student, 19 years old

- Suggested adding extra options to the three-dot menu in the photo zoom interface.
- Recommended adding emphasis to the gallery for better navigation.
- Rated usability as **7/10**, describing the design as pleasant.
- Reported ease in completing most procedures, with only minor difficulties.

#### 4.6.0.3 Fine Arts Student, 23 years old

- Experienced no difficulty performing any tasks using the app.
- Described the app as **easy to use**.
- Suggested improving the art direction for a more cohesive aesthetic.
- Recommended enhancing the app's textual content for better communication.

## 4.7 Trial and Error

Trial and error played a pivotal role throughout the development of the eShadow LCE platform, particularly when determining the optimal background removal model, technologies, and deployment methods. The selection of the background removal model required rigorous testing of multiple pre-trained U-Net models to balance performance and computational efficiency. For instance, while larger models like the standard unet.onnx provided high accuracy, they proved too resource-intensive for real-time processing on mid-range systems. By systematically evaluating models of varying sizes and architectures, the team identified the unetp.onnx model as the optimal choice due to its exceptional performance and minimal computational demands, making it suitable for local execution on general-purpose laptops. This process exemplifies how trial and error, combined with performance benchmarking, ensured the final selection aligned with the project's practical and technical constraints.

The iterative development of the style transfer server further highlights the importance of trial and error. Initial implementations of the Stable Diffusion server involved direct use of pre-trained models, which lacked the nuanced traditional Greek shadow theater aesthetic required for the project. Through multiple iterations, a curated dataset of 12 representative images was compiled, and the LoRA fine-tuning technique was employed to train a custom model. Experimentation with hyperparameters, such as learning rates and epoch counts, was critical to achieving high fidelity in the output while maintaining computational efficiency. Similarly, deploying the server involved trials with various configurations to optimize GPU

usage and minimize response latency, ensuring that even resource-intensive style transformations could operate seamlessly in real-time scenarios. This iterative process of testing, feedback, and refinement was integral to achieving the desired quality and performance in the platform's final implementation.

# 5. Detailed System Functionality

This chapter provides an in-depth showcase of the functionality offered by eShadow LCE (Live Collaboration Edition) presented in a step-by-step manner. It also includes UML activity diagrams wherever needed to further clarify the processes.

## 5.1 Taking a Picture

When presented with the start screen the user presses the camera button to take a picture.

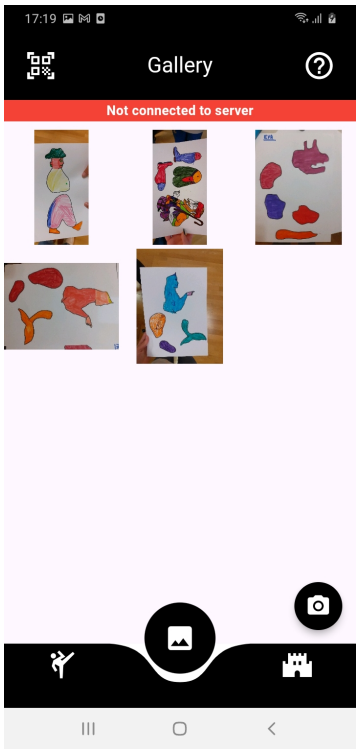
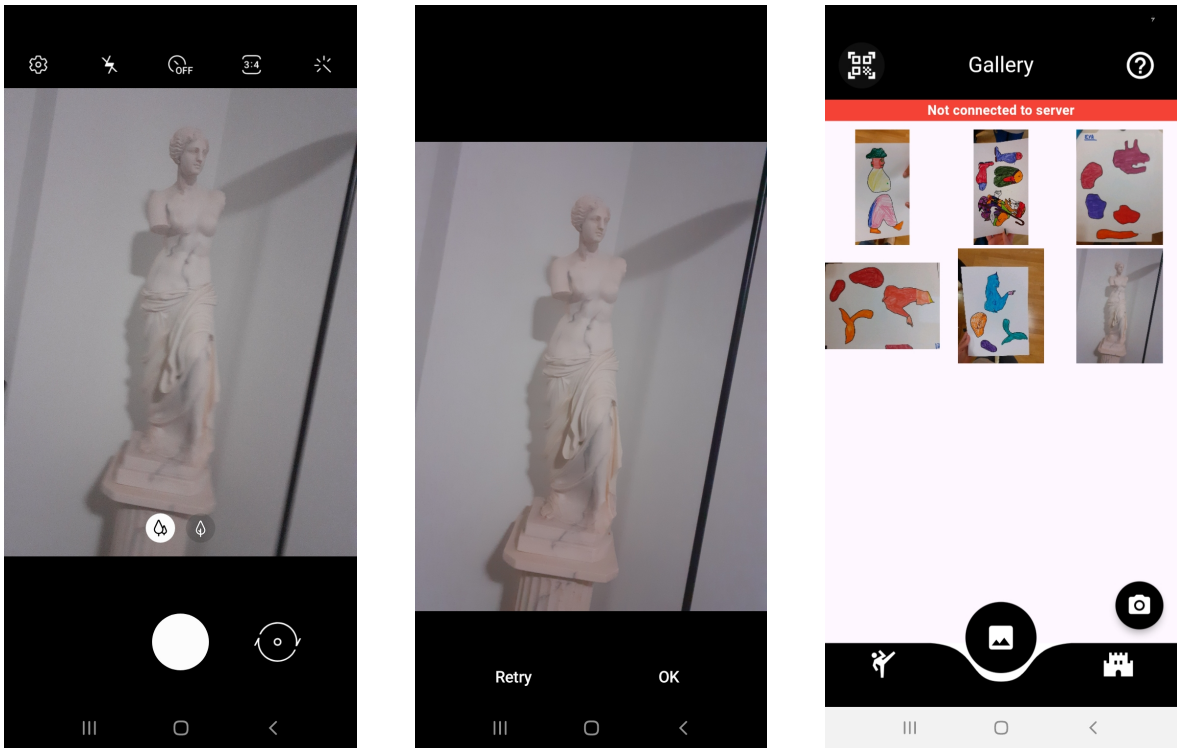



Figure 5.1: Application Start Screen




(a) Camera screen (b) Confirm Capture (c) Image Added to Gallery

Figure 5.2: Taking a Picture

## 5.2 Applying Style Transform to an Image

In order to apply style transformation to an image one must first setup the style transform server. To run the server efficiently, a CUDA core [43] compatible machine is highly recommended. More details on how to setup this server will be released once the final code is open-sourced, for now, the server is running through it's development environment.

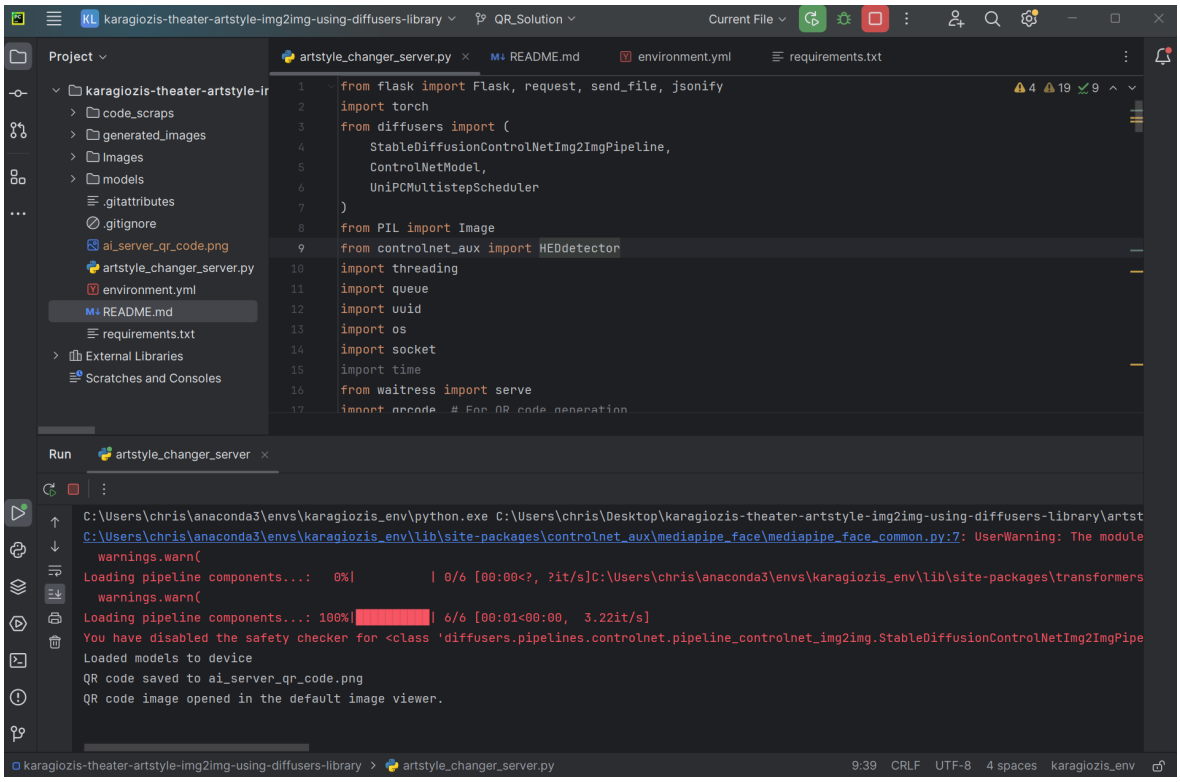



Figure 5.3: Instance of the Server Running on Pycharm IDE

Since this specific instance of the server is designed to run on the local network, when it starts up, it creates a QR code representing its local address for mobile apps to scan and connect.

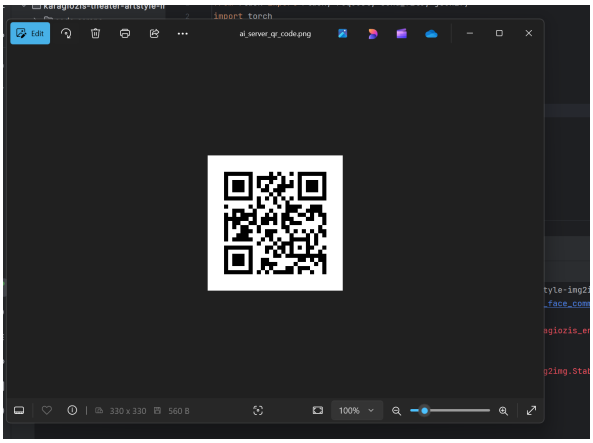


Figure 5.4: QR Code Containing Application Local IP and Port

The user can connect to the style transform server by scanning the generated QR code. After the application connects successfully to the server, a connected status indicator is presented on the top of the screen. The user now long-presses on the image to be style transformed to bring up the menu, then chooses the option `Artstyle Transform`. The application sends the job to the server and waits for completion. A loading indicator will be presented while waiting.

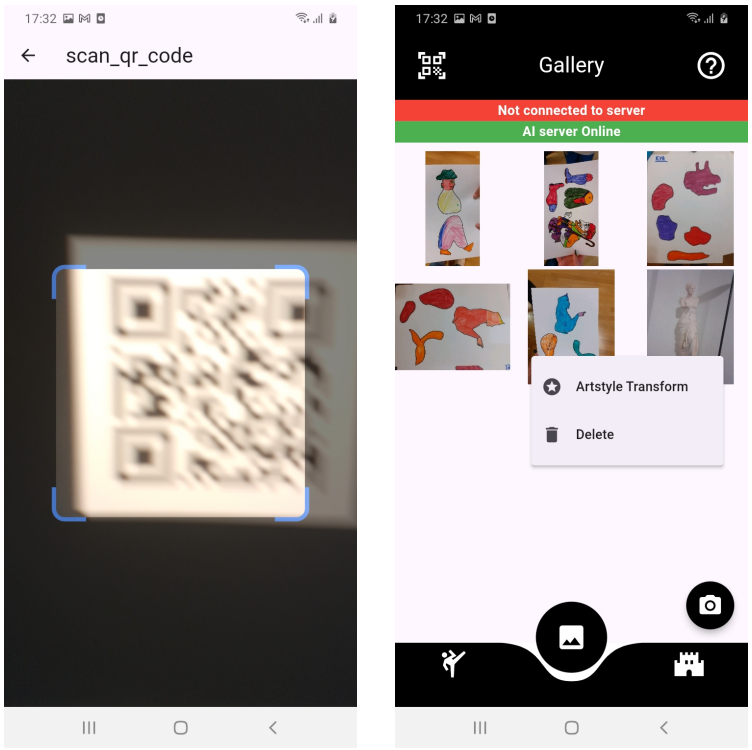



Figure 5.5: Connecting to Style Transform Server and Initiating Style Transform process

When the server receives a request, it queues it and when the time comes processes if. Depending on the hardware, processing time can vary, however the optimizations made aim for it to be in the magnitude of a couple of seconds. The example shown below is run on an RTX 4060 8GB Vram PC and takes about 2 seconds.

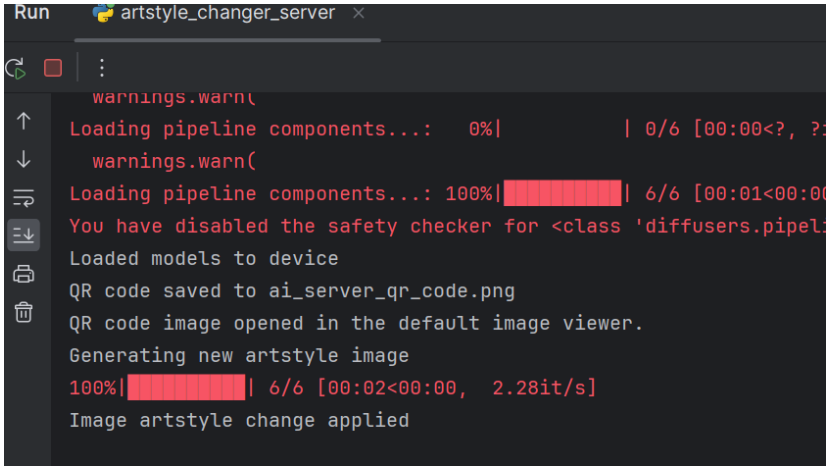
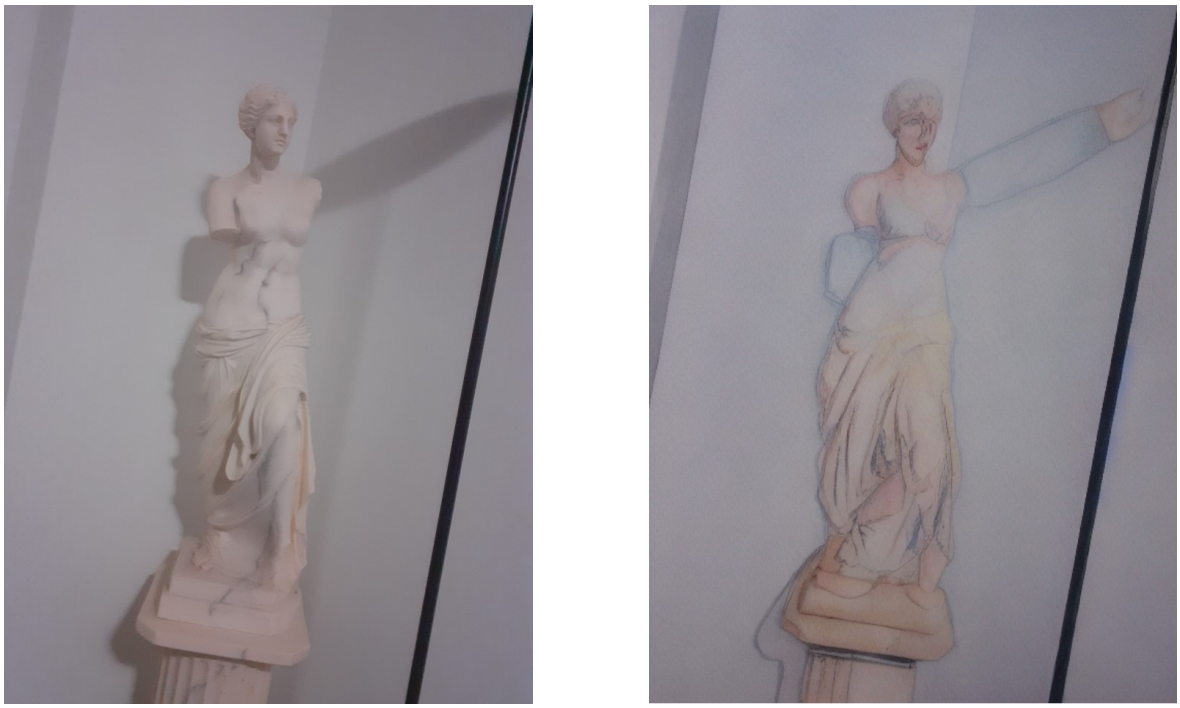



Figure 5.6: Example of the Server Running

Depending on the lighting and the complexity of the image, the results may vary. However, in most cases, the results are at least adequate. For instance, in the example below, we have intentionally used a poorly illuminated image with confusing shadows to present an example close to real world use.



(a) Initial Image (b) Output Image

Figure 5.7: Result Comparison

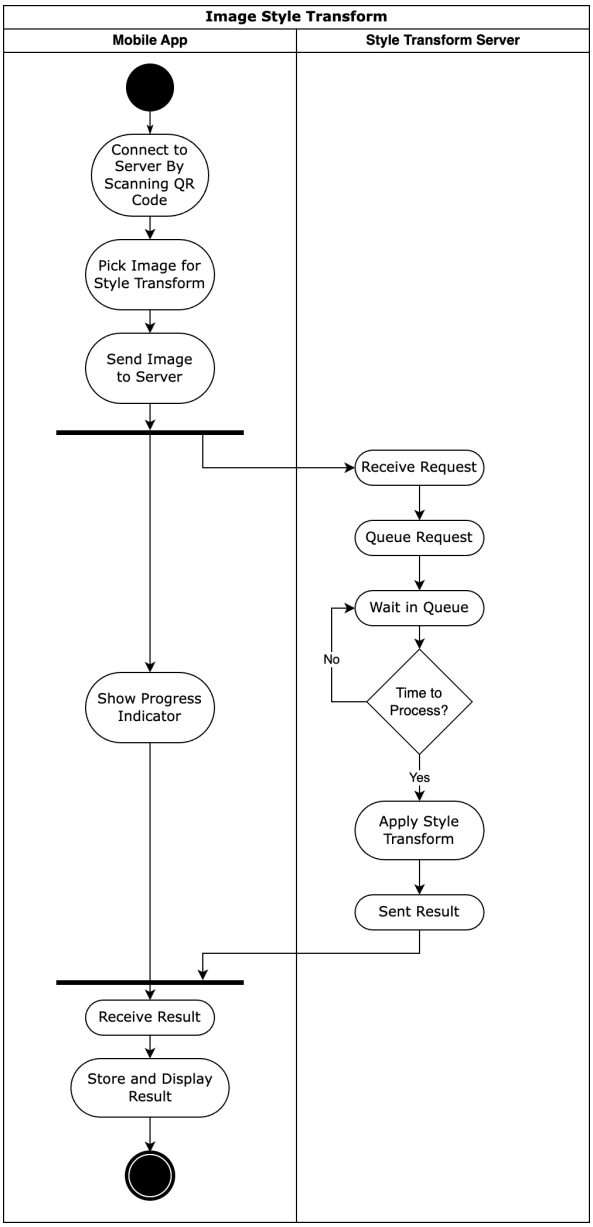



Figure 5.8: Image Style Transform Activity Diagram

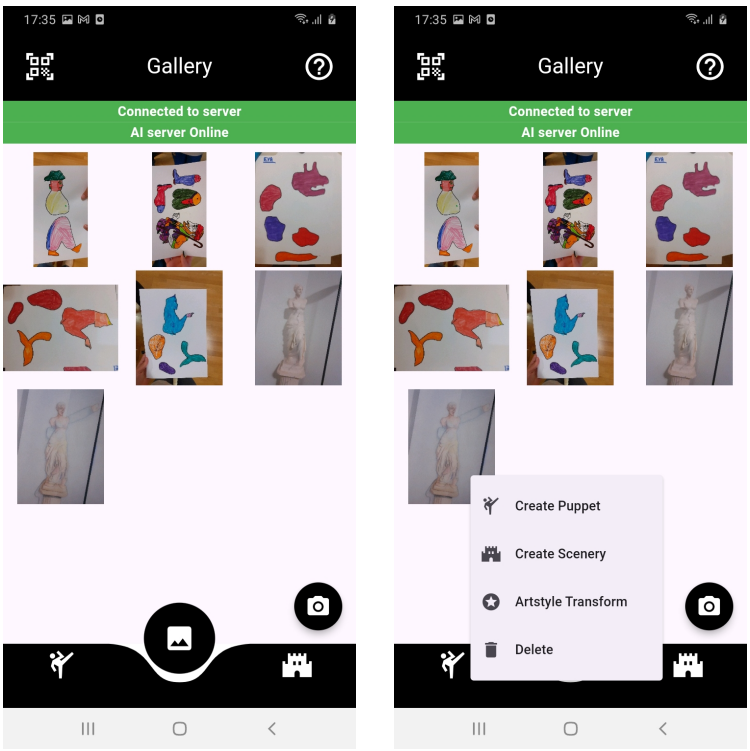
## 5.3 Figure Creation

The system can create figures from any type of real world object or even people. The system also includes a separate flow for the case that the parts are already separately drawn that takes advantage of some assumptions to further simplify the process. Below we present in detail the two flows available to create a figure.

### 5.3.1 Figure From Real World

An image of an object or a person is enough for the system to turn it into a figure. Style transformation is not mandatory to initiate this process, however having started up the main

server and connected to it using the QR code scanner is required, due to the design decision to run resource demanding processes on the main computer.




(a) User Has Connected to Main Server

(b) Long Press on the Image and Initiate Puppet Creation

Figure 5.9: Initiating Puppet Creation

After initiating the creation process the corresponding flow is selected from the options presented.

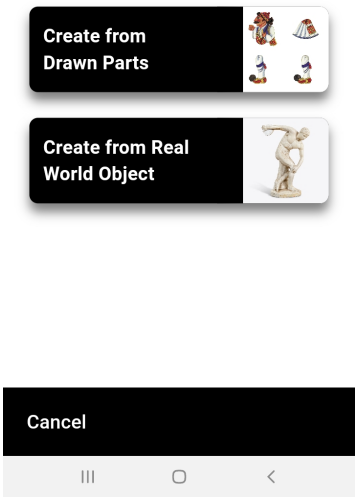



Figure 5.10: Choose Creation Flow Options

In the First step, the option to crop the image is presented to remove unnecessary detail.

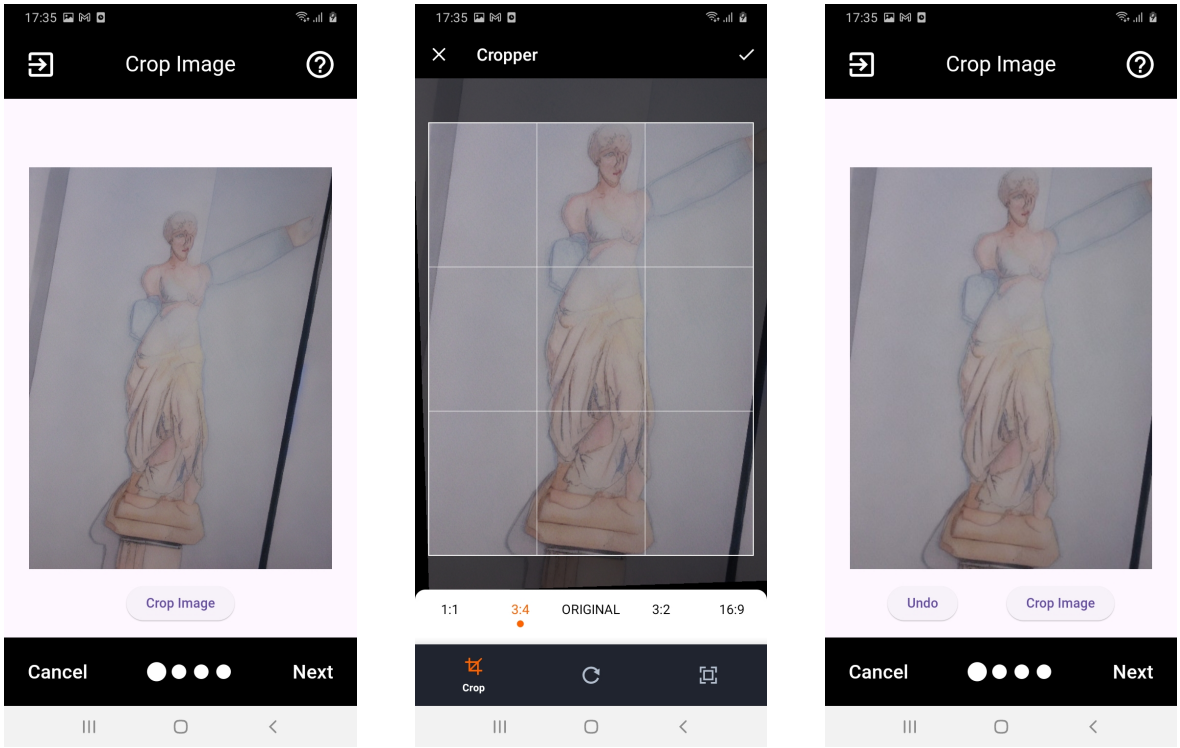
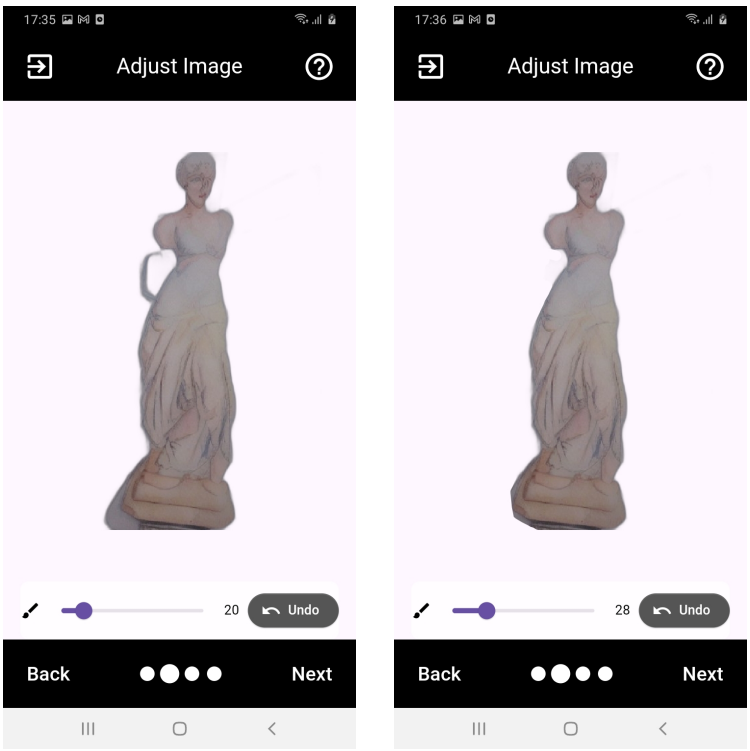



Figure 5.11: Image Cropper

After the image is cropped, it is sent automatically to the main server for background removal, this process usually takes no longer that a few seconds even on low-spec machines.

After the background is removed, the user is given the option to remove imperfections left from the background removal process by hand.

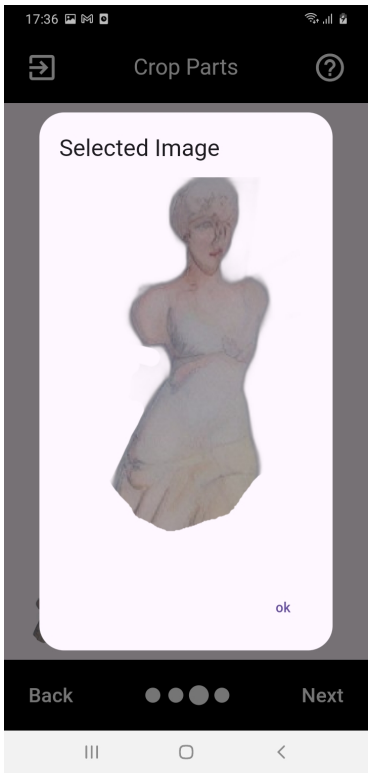



(a) Initial Image (b) Imprefections Erased

Figure 5.12: Erase Background Removal Leftovers

On the last step the user must circle using touch gestures around desired areas to create distinct parts. Created parts are presented below the image. There also the option to delete parts.

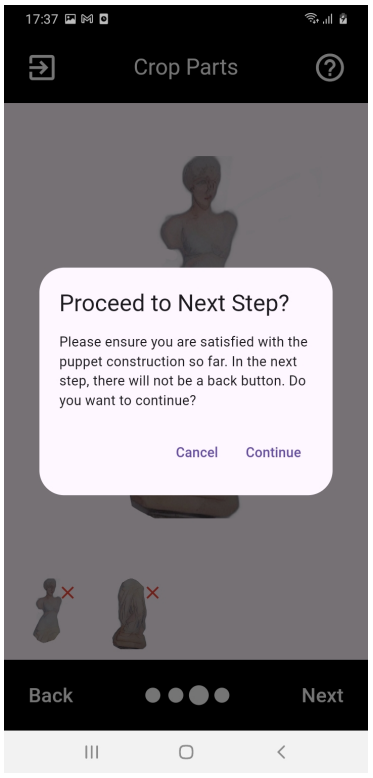

(a) Selecting Figure Torso Area




(b) Selection Confirmation


(c) Selecting Figure Leg Area


(d) Selection Confirmation

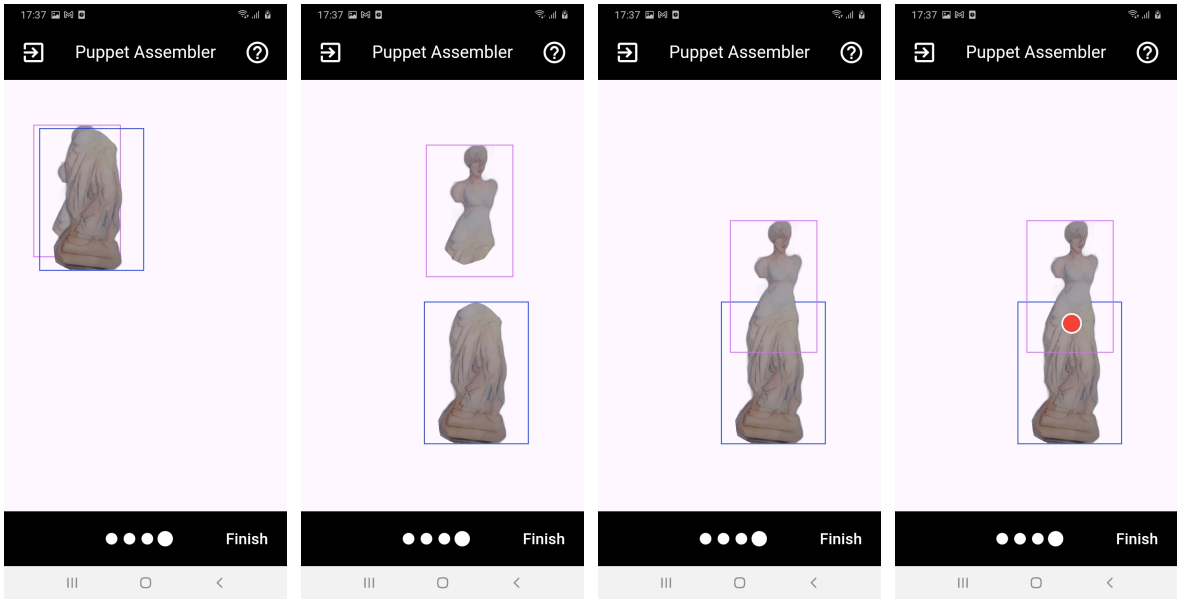



(e) Confirmation Before Last Step

Figure 5.13: Defining Figure Parts

The last step is the assembly screen. here the user is presented with the parts selected in

the previous step. Those parts can be moved around and set at desired positions. Through long press, joints between parts can be defined. To remove a joint a user can simply tap on it.




(a) Assembly Screen (b) Moving Parts (c) Combining Parts (d) Defining Joints

Figure 5.14: Figure Assembly

After the figure creation is over the created figure can be accessed through the Puppets tab.

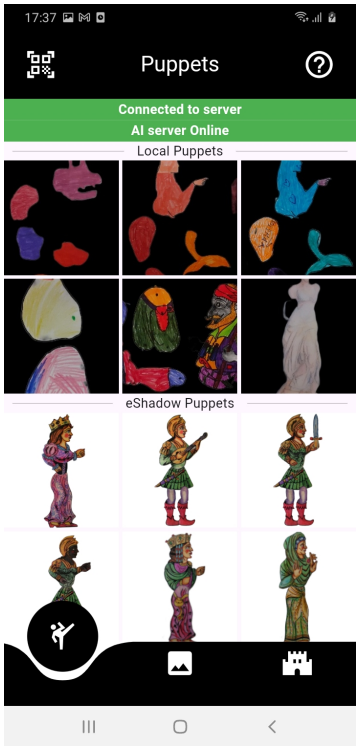



Figure 5.15: Puppets Tab

Long-pressing on the created figure presents available options, including loading it tho the screen.

Figure 5.16: Created Figure Loaded to Digital Stage

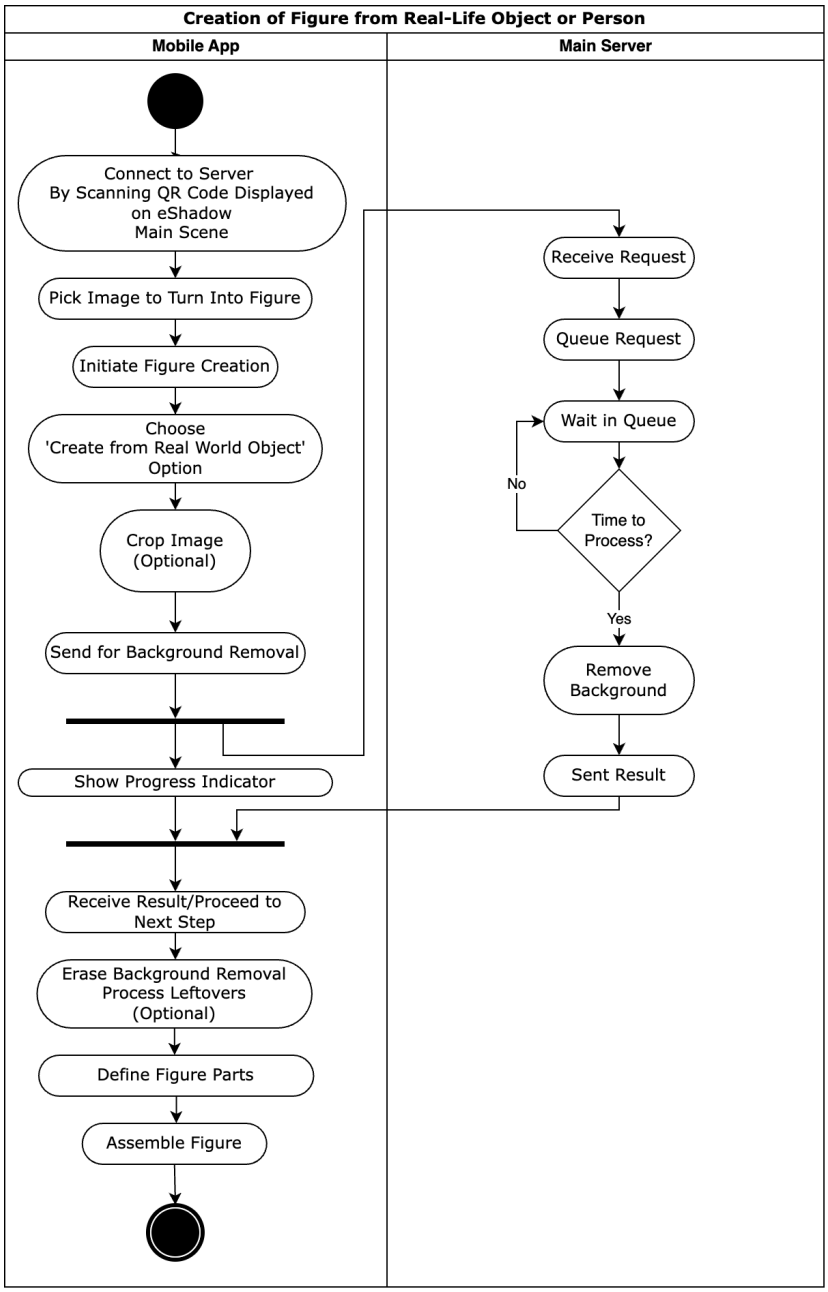


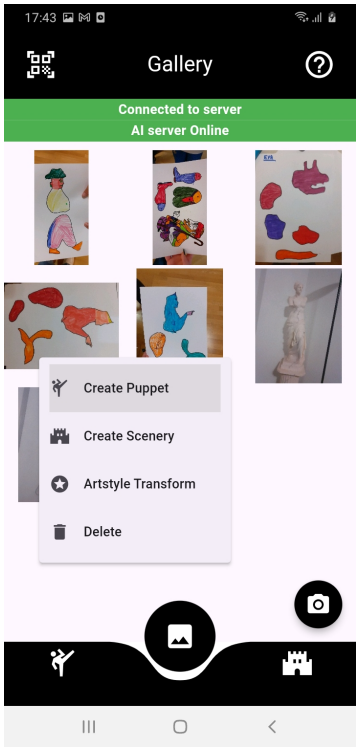

Figure 5.17: Figure Creation From Real World Activity Diagram

## 5.4 Puppet From Drawn Parts

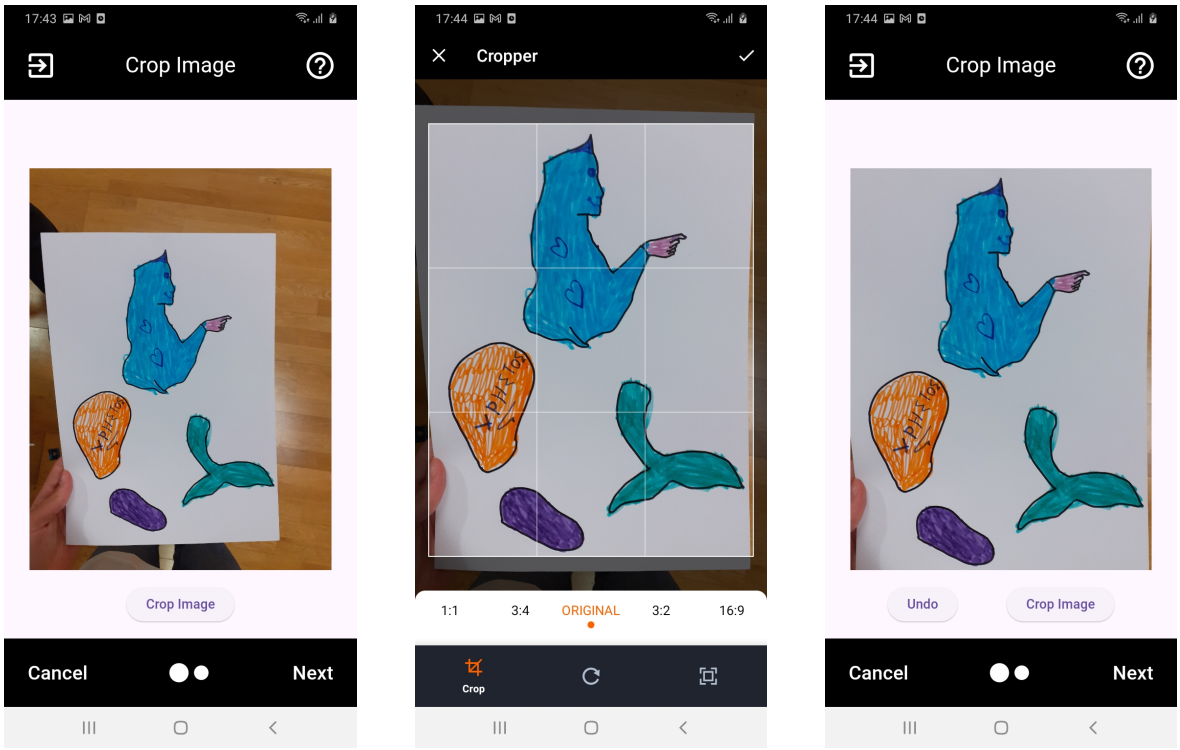
A simpler workflow has been implemented for the case that the figure is already in separated parts.


Figure 5.18: Drawn Figure With Separated Parts

The user long-presses on the drawing image and initiates the corresponding flow,



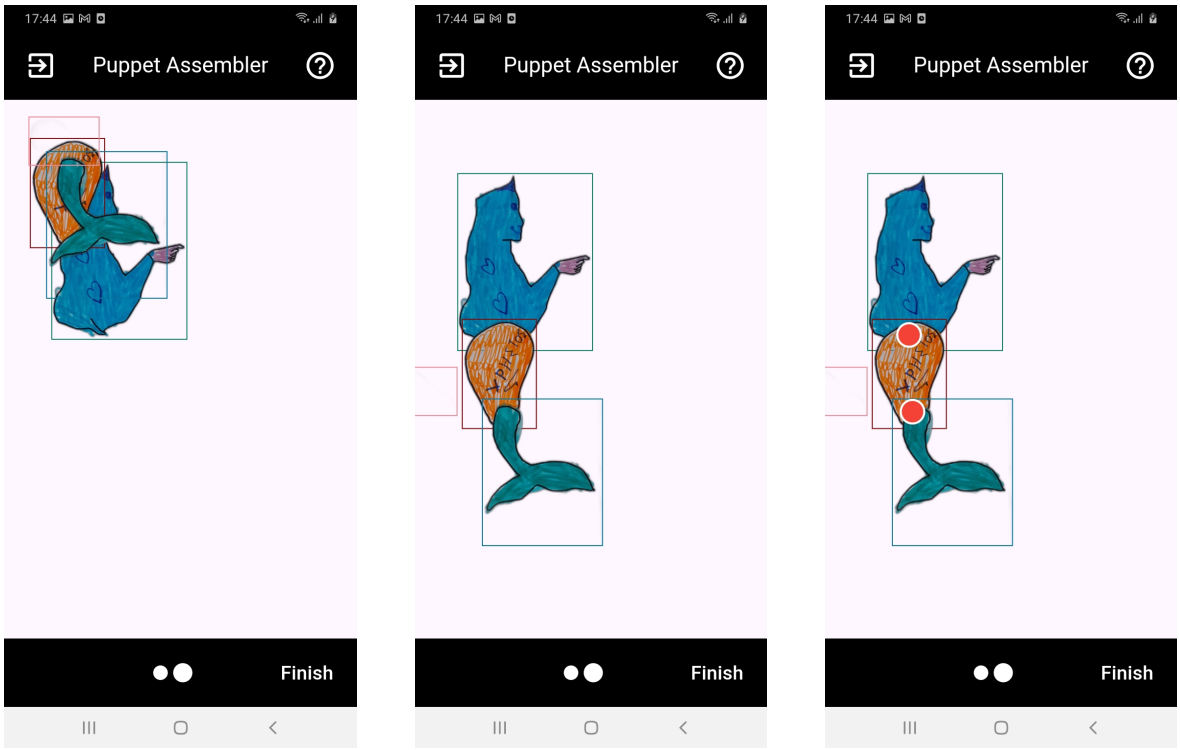

In the crop step, it is advised to crop out the external environment and maintain a clean instance of the parts.




(a) Image Before Crop (b) Image During Crop (c) Image After Crop

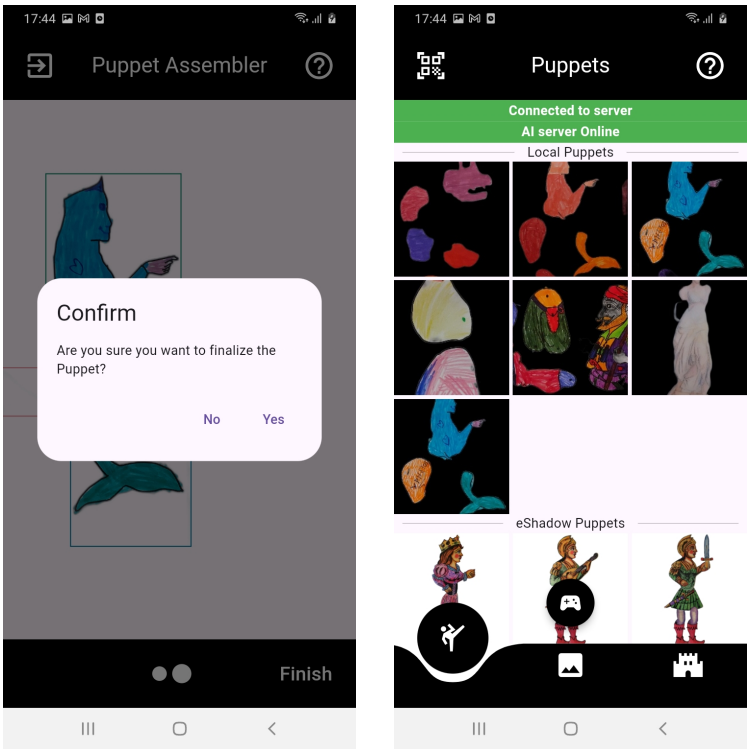
Figure 5.19: Cropping the Image Correctly

After the crop step has finished, the image is sent to the server to have it's background cleared and the parts separated. The separated parts are presented on the figure assembly screen.




(a) Figure Assembly Screen (b) Moving Parts Into Position (c) Defining joints

Figure 5.20: Figure Assembly Process




(a) Confirm Assembly (b) Newly Created Figure

Figure 5.21: Finalize Creation

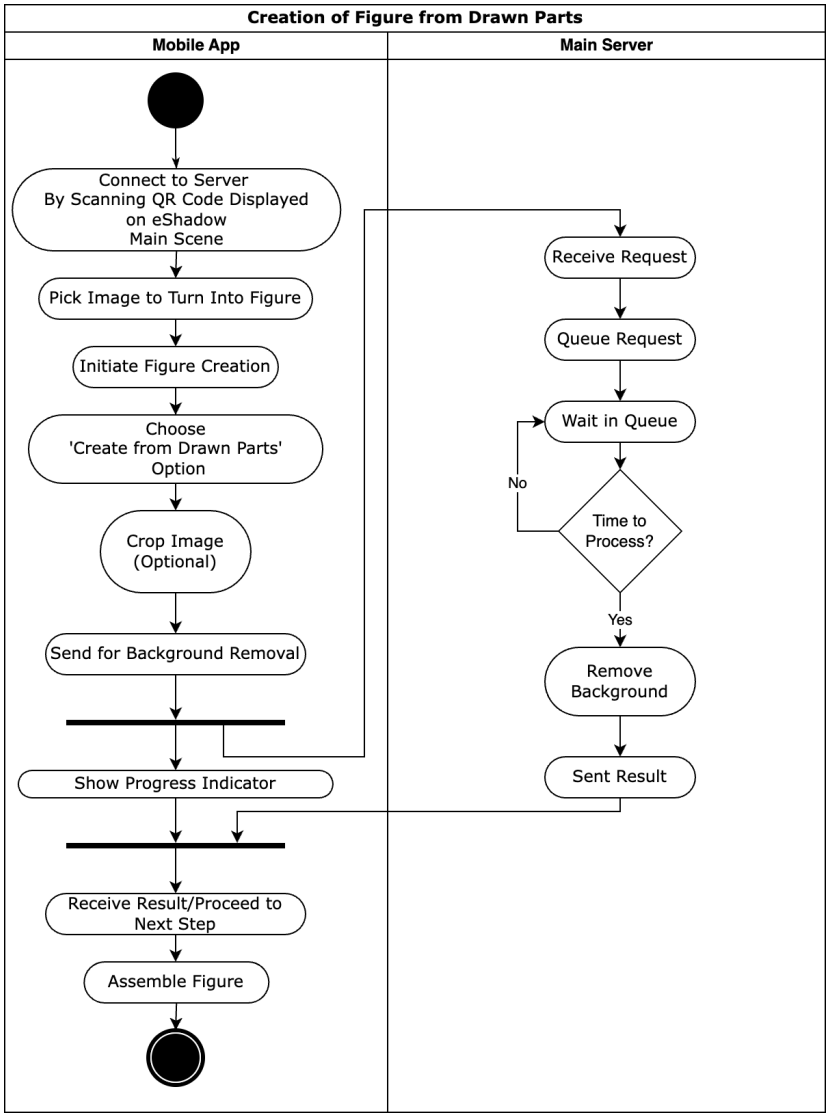



Figure 5.22: Figure Creation From Drawing Activity Diagram

(a) Figure Loaded to Screen 1 (b) Figure Loaded to Screen 2

Figure 5.23: Showcase of Created Figure

## 5.5 Background Creation

Background objects can also be created through the app, the process is extremely straightforward. The user chooses the input image by long-pressing, applies crop if nessesary and finaly the image is cleared of its background and store as a background object in its respective tab.

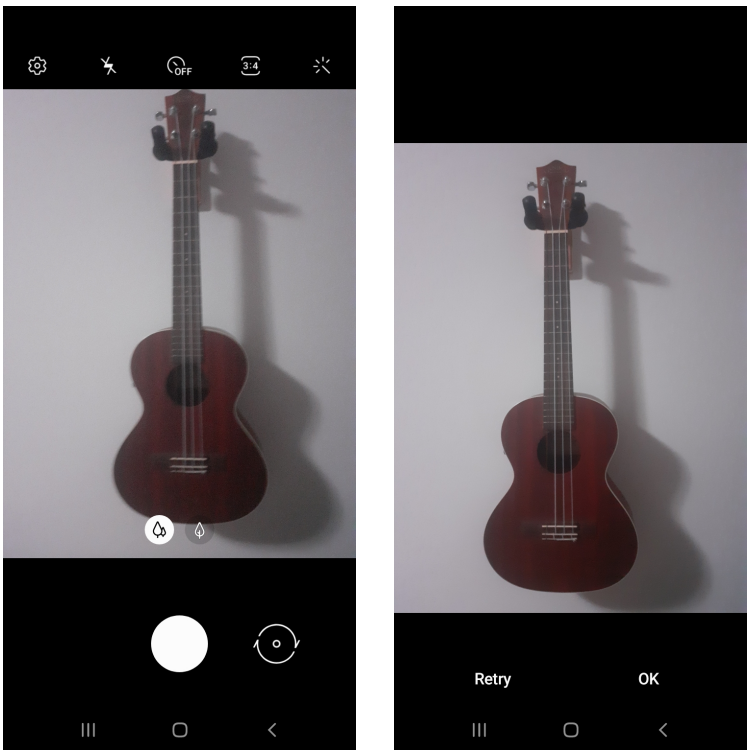



Figure 5.24: Taking Image to Make Background Object

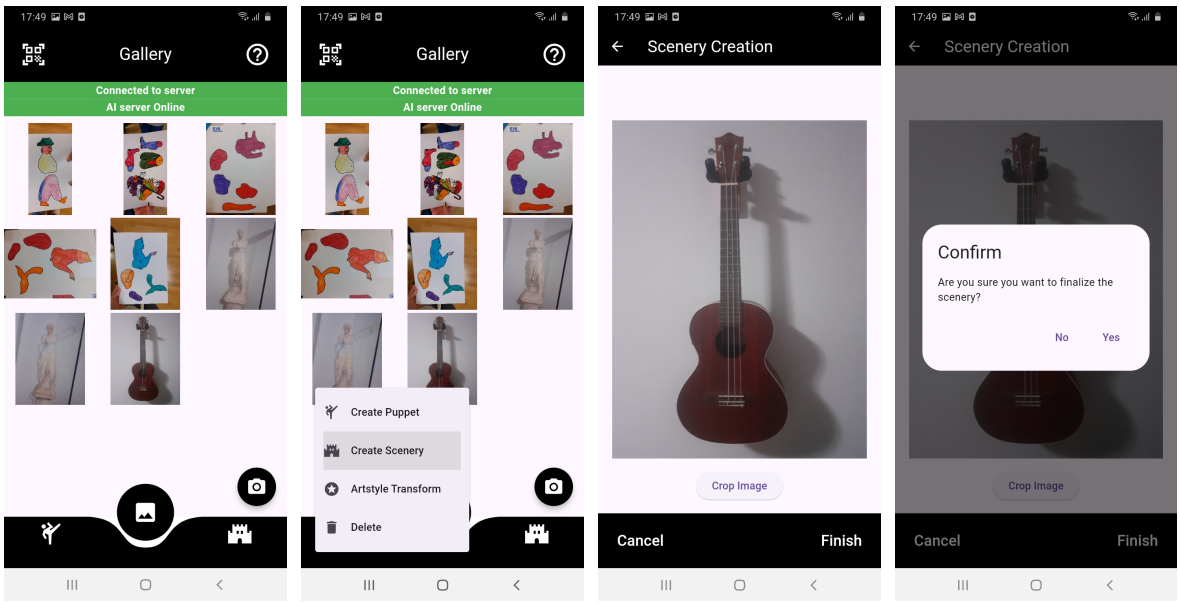



Figure 5.25: Background Object Creation Process

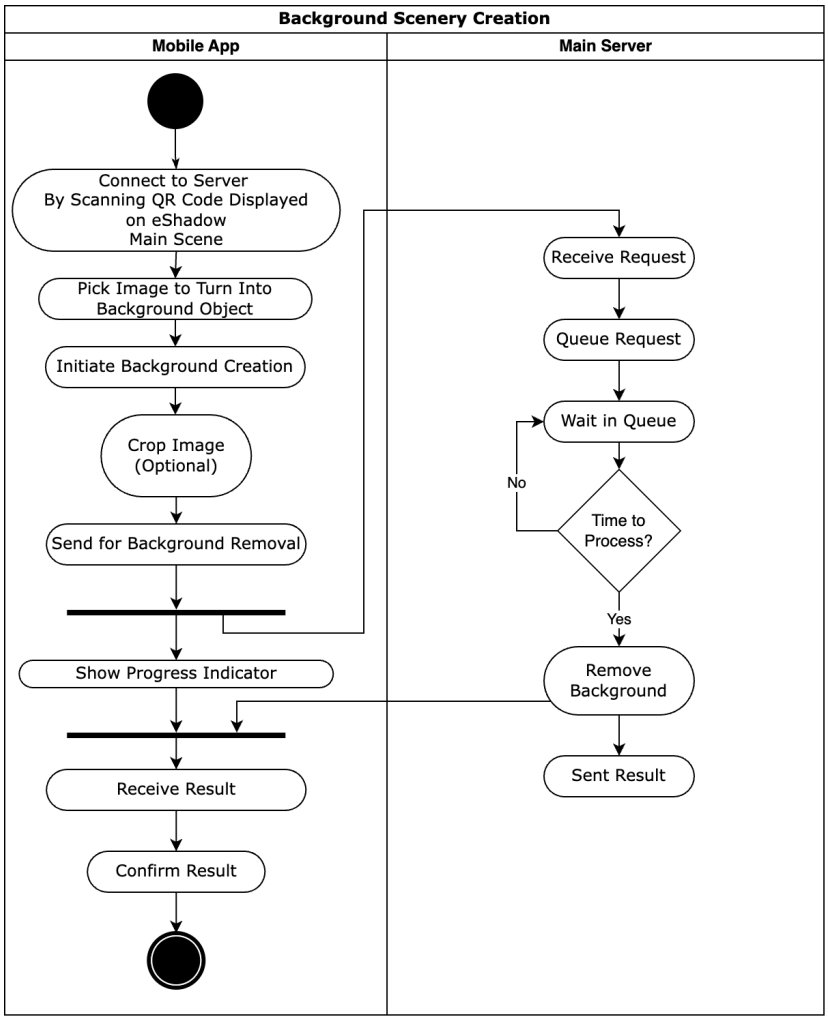



Figure 5.26: Background Scenery Creation Process Activity Diagram

## 5.6 Real Time Control

A figure/background object can be controlled and manipulated using the dedicated controller screen. To Initiate control, the user long-presses on the desired object and chooses the `Load to Stage` Option. After the option is pressed, the selected object is instantiated on the eShadow stage and the controller screen appears on the mobile device.

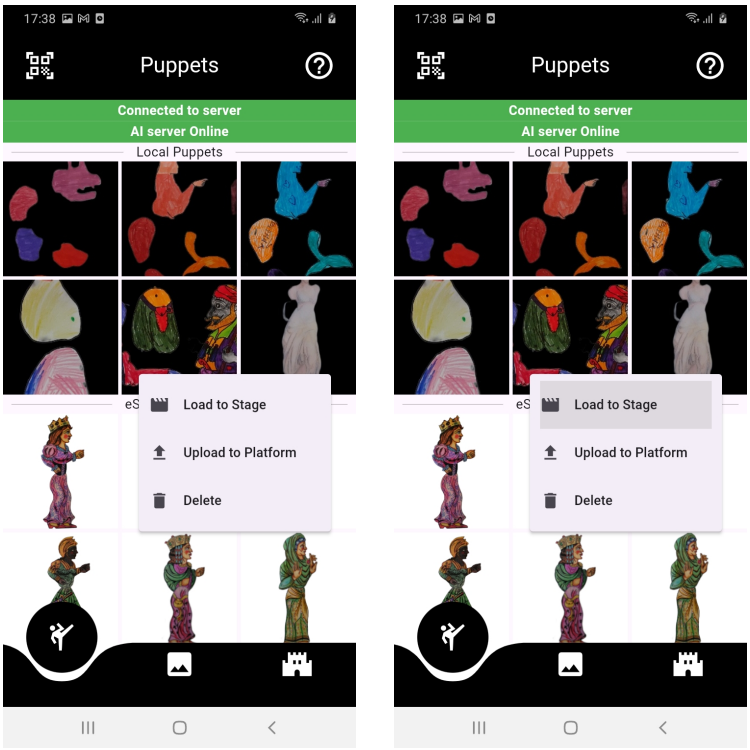



Figure 5.27: Initiating Puppet Control

Object Instantiated on the eShadow Stage

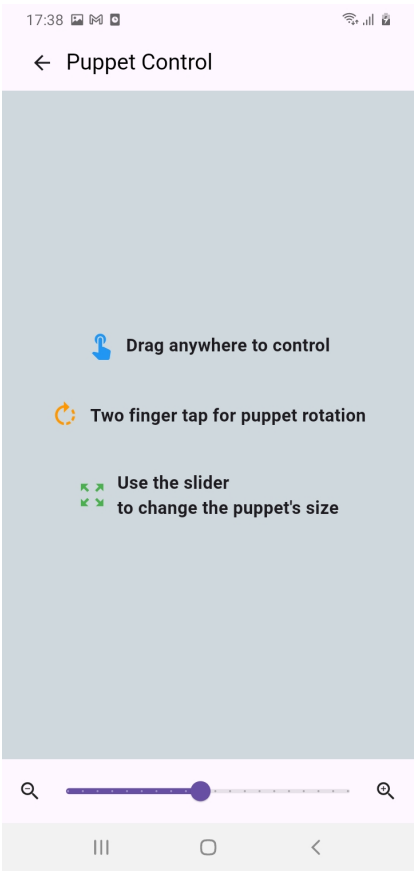



Figure 5.28: Figure Controller

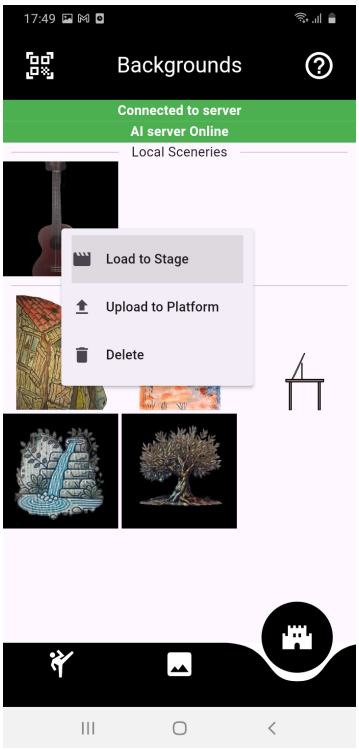
The user can drag anywhere using touchscreen finger-drag to move the object in real time.

Figure 5.29: Figure Response to Movement

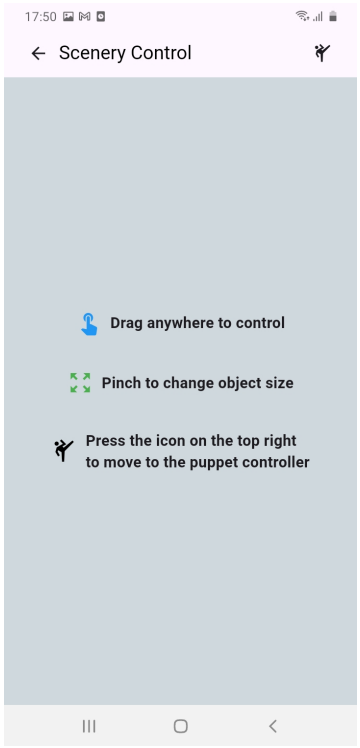
There is also the ability to rotate the figure, or change its size.

Figure 5.30: Figure Real-Time Size Manipulation

Similar rules apply to background manipulation, with main difference the inability to rotate.




(a) Long Press to Bring Down Menu




(b) Controller Interface

Figure 5.31: Scenery Object loaded to Screen

## 5.7 Hints

In order to assist the user, a versatile hint button has been implemented, providing the user with relevant hints to the current running process.

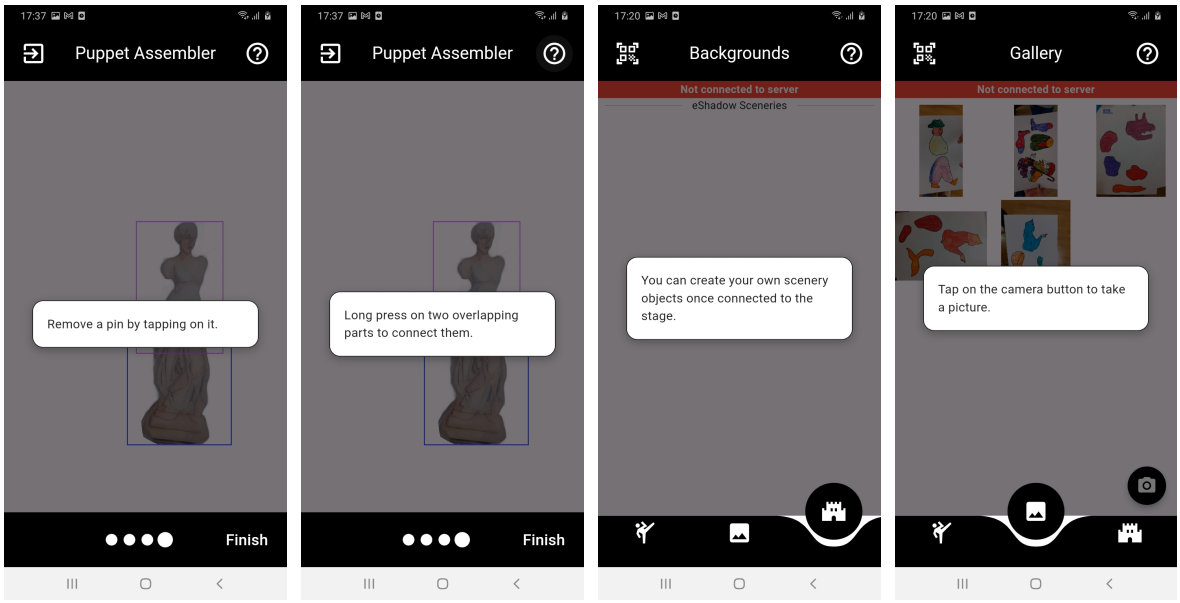



Figure 5.32: Hints Sample 1

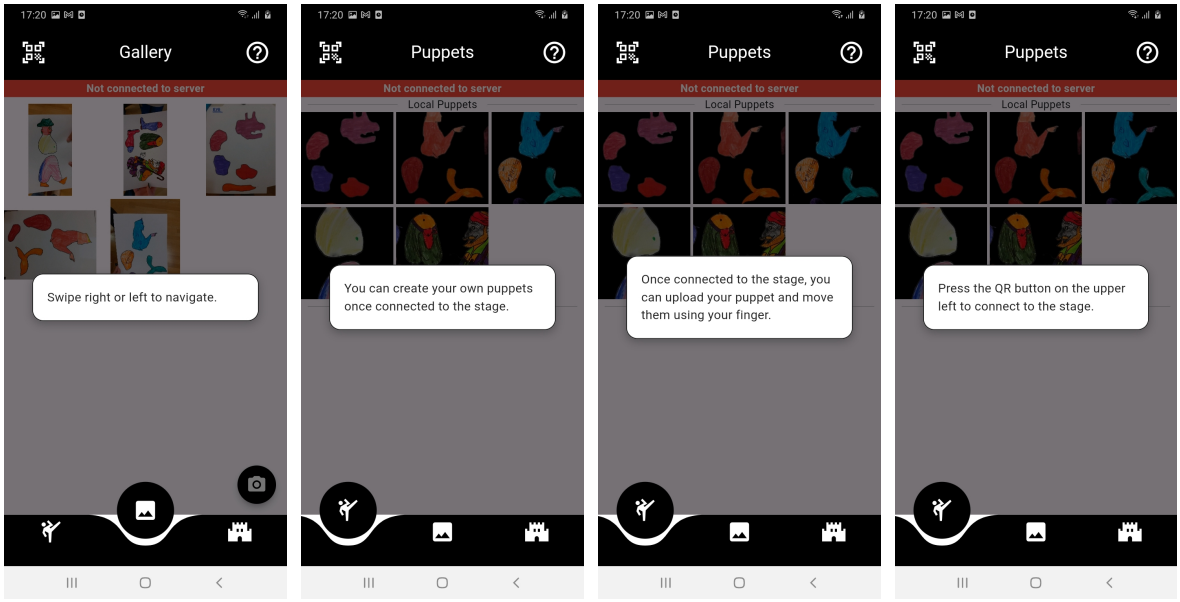



Figure 5.33: Hints Sample 2

# 6. Implementation

This chapter provides an overview of the implementation of the eShadow Live Collaboration Edition (LCE) system. It starts with a high-level description of the system and its components, as well as the communication methods between them. The chapter then delves deeper into each component, presenting crucial pieces of code that help understand the overall functionality of the system. In the end, some of the most important design decisions are presented and discussed.

## 6.1 System Overview

The System comprises multiple components interacting over a local network, spanning various devices and designed with modularity and ease of deployment in mind. Below, a diagram presents the system's primary components, their interactions, and other relevant details.

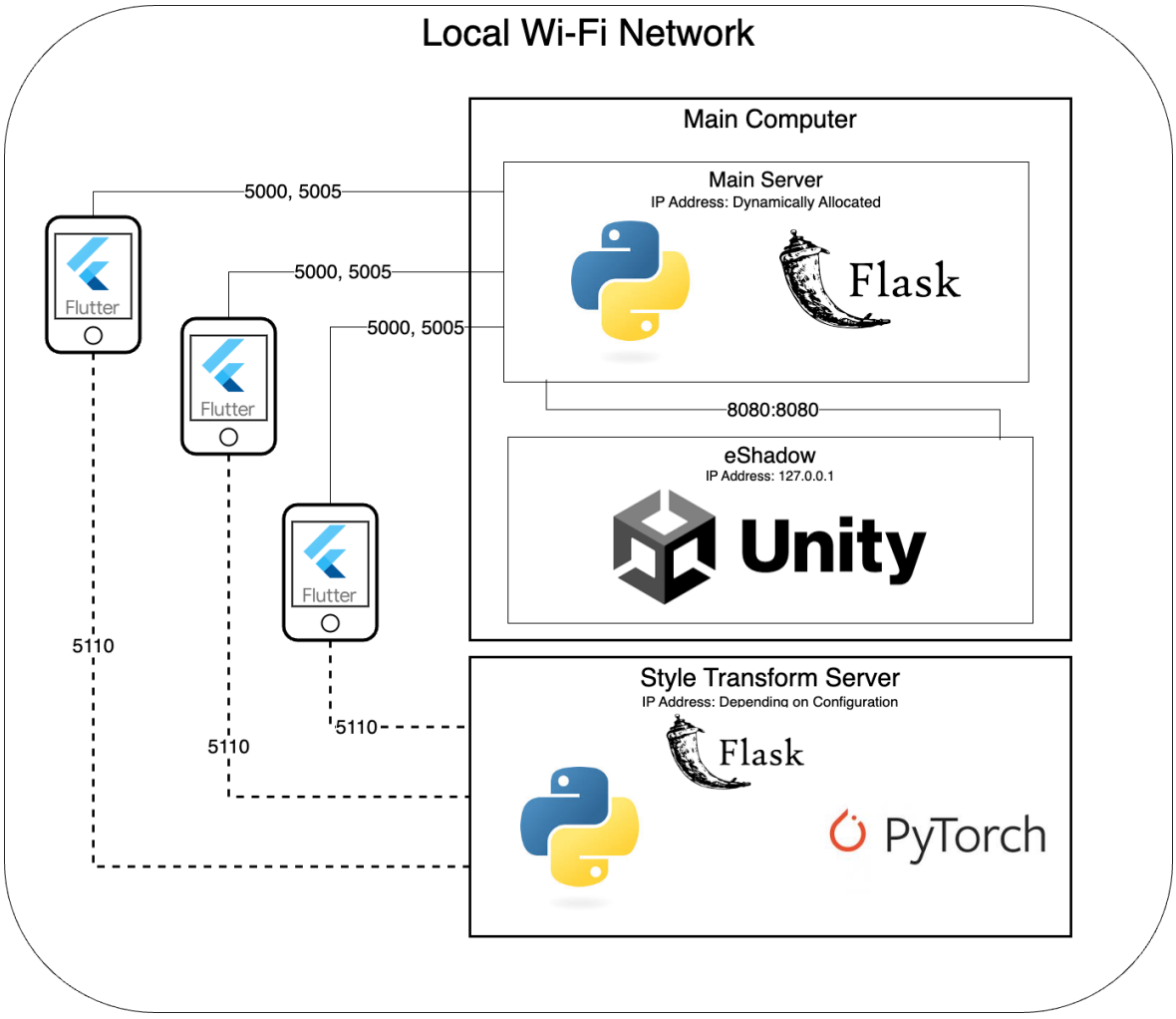



Figure 6.1: High Level Overview of the System Architecture

### 6.1.1 System Main Components

- **eShadow Companion (Mobile Application) :** Built with Flutter [21], aiming to maximize compatibility with the various mobile devices out there, the app serves as the main interface for users and provides access to all system core functionality. It acts in many ways similar to a frontend client application; however, it stands out by leveraging the mobile device's hardware (Hi-Res camera, touchscreen) as well as performing a great deal of processing tasks directly on the device.

- **Main Server:** Written in the Python programming language [49] using the Flask library [46] to facilitate communication, the main server acts as the primary gateway to the system's functionality, while also facilitating interaction between the mobile application and the enhanced eShadow platform. This server leverages Python's robust ecosystem of libraries to perform advanced image processing tasks as well as hosting an AI model for background removal.

- **Style Transform Server:** Written also in the Python programming language [49] using

the Flask library [46] to facilitate communication, this server integrates the advanced PyTorch framework [15] to deliver machine learning capabilities, incorporating a combination of the Stable Diffusion model [51], ControlNet [68], and a custom LoRA [25] model trained on traditional Greek shadow theater figures. The server provides a straightforward REST API, with a primary endpoint that accepts an input image, applies a style transformation, and returns the transformed image as output. Due to its resource-intensive nature, the system is designed to run efficiently on mid-to-upper-tier computers equipped with CUDA-supported [44] GPUs. Built with modularity in mind, the server ensures that even in its absence, the system remains fully functional, albeit with one less feature.

- **eShadow LCE (Unity Platform):** eShadow LCE serves as a virtual stage for real-time puppet manipulation and rendering. While preserving all the original eShadow functionality, it has been upgraded to integrate seamlessly with other components of the application. Additionally, it incorporates several quality-of-life enhancements designed to enhance the platform's versatility and improve the overall user experience.

Communication between these components is facilitated through RESTful APIs and UDP for some low-latency operations.

### 6.1.2 Intra-Component Communication

The communication between components occurs within a local network where the corresponding IP addresses are shared using QR codes. To address functionality, the message is sent as the specific component's IP address at the function's pre-designated port number, utilizing HTTP or UDP protocols, depending on the operation. Below, we offer a detailed explanation of each port's roles, functions, and the communication protocol it employs within each component.

- **Main Server:**
  - **Port 5000 (HTTP):** Handles REST API requests from the Mobile Application for operations like background removal, segmentation, and puppet/scenery uploads.
  - **Port 5005 (UDP):** Receives low-latency control messages from the Mobile Application, which are relayed to the Unity platform.
  - **Port 8080 (HTTP):** Used to communicate REST API requests as well as UDP messages to the Unity platform.
- **AI Server:**
  - **Port 5110 (HTTP):** Processes AI-based style transformation requests from the Mobile Application.

- **Unity Platform:**
  - **Port 8080 (UDP, HTTP):** The Main Server sends low-latency commands for puppet and scenery control to the Puppet Server, as well as figure/scenery related data.

## 6.2 Frontend (Mobile Application)

Developed using Flutter, the mobile application serves as the primary user interface for interacting with the eShadow LCE system.

### 6.2.1 Key functionalities

- **Connectivity:** The app utilizes the phone's camera to read connectivity information displayed by other components as QR codes, enabling communication with them.
- **Image Capture:** Users can capture images, store them in the app, and apply various processes to them.
- **Gallery Management:** Enables users to manage created puppets, sceneries, and taken photos and apply various operations on them, such as deleting, loading to eShadow scene, etc.
- **Figure and Background Creation:** Enables users to transform captured images into figures or backgrounds that can be loaded, shared, and interacted with on the enhanced eShadow platform.
- **Art Style Transformation:** Facilitates communication with the style transform server to apply a style transform to the selected image and convert it into traditional Greek shadow theater aesthetics.
- **Real-Time Control:** Provides an intuitive interface for manipulating figures and backgrounds in real-time, with touch gestures such as dragging, resizing, and rotating.

### 6.2.2 Flutter Dependencies

The project utilizes various Flutter dependencies to enable key features such as navigation, image processing, network communication, and state management. Below is a detailed description of each dependency and its purpose, also including the respective citations as of the source of each one:

#### 6.2.2.1 Core Flutter Dependencies

- **flutter:** Provides the core Flutter SDK for building user interfaces and managing application behavior.
- **provider:** A popular state management library for handling application state efficiently [10].

#### 6.2.2.2 Navigation and UI Enhancements

- **smooth_page_indicator:** Adds elegant page indicators for page views [47].
- **curved_navigation_bar:** Implements customizable and aesthetically pleasing curved navigation bars [63].

#### 6.2.2.3 Image Processing and File Handling

- **image_picker:** Enables users to pick images from the gallery or camera [54].
- **image:** Provides advanced image manipulation features like resizing and encoding [9].
- **flutter_image_compress:** Compresses images to reduce file size while maintaining quality [45].
- **image_cropper:** Allows users to crop images before further processing [27].
- **archive:** Supports archiving and unarchiving files, such as handling ZIP files [14].
- **path_provider:** Provides paths to application directories on the device's filesystem [55].

#### 6.2.2.4 Networking and Communication

- **http:** Facilitates HTTP requests for REST API communication [53].

#### 6.2.2.5 Barcode Scanning and Unique Identification

- **flutter_barcode_scanner:** Adds functionality to scan barcodes and QR codes using the device camera [24].
- **uuid:** Generates unique identifiers for data management and object identification [23].

#### 6.2.2.6 Storage and Preferences

- **shared_preferences:** Provides persistent storage for simple key-value pairs, useful for saving user preferences and app settings [56].

### 6.2.3 Important Code Breakdown

#### 6.2.3.1 QR Code Connection and Server Interaction

The application integrates a QR code scanner to enable users to connect to servers dynamically. Upon scanning a QR code, the application parses the server address, verifies connectivity, and updates the status for both the main and AI servers. The fundamental principle of the code presented below is that it scans the QR code using the camera and attempts to establish communication with the server indicated. It determines the type of server based on the port it reads. If communication is successful, it updates the application's status. If it's the main server, it receives and stores a figure ID and a scenery ID, which are used to reference figure and background items in subsequent operations.

```
Future<void> scanQRCode() async {
  // Navigate to the QR code scanner screen and wait for the result
  final qrCode = await Navigator.push(
    context,
    MaterialPageRoute(builder: (context) => QRCodeScannerScreen()),
  );

  if (qrCode == null) return; // User canceled the scan

  final address = qrCode.toString();
  try {
    // Parse the URL to get the port number
    Uri uri = Uri.parse(address);
    ...
    ...
    ...

    // Construct the URI with the appropriate endpoint
    final serverUri = uri.replace(path: endpoint);
    // Initiate communication with the server and receive User ID's
    final response = await http.get(serverUri);

    if (response.statusCode == 200) {
      final jsonResponse = json.decode(response.body);
      setState(() {
        if (port == 5000) {
          // Main server
          global_variables.connectedToServer.value = true;
          global_variables.qrData = uri.origin;
          if (global_variables.puppetID == "" ||
              global_variables.sceneryID == "") {
            global_variables.puppetID = jsonResponse['id1'];
            global_variables.sceneryID = jsonResponse['id2'];
```

```
          }
        } else if (port == 5110) {
          // AI server
          global_variables.aiServerAddress = uri.origin;
          global_variables.connectedToAIServer.value = true;
        } else {
          // Unknown server
          ScaffoldMessenger.of(context).showSnackBar(
            ...
          );
        }
      });
    } else {
      setState(() {
        if (port == 5000) {
          global_variables.connectedToServer.value = false;
        } else if (port == 5110) {
          global_variables.connectedToAIServer.value = false;
        }
      });
    }
  } catch (e) {
    ...
  }
}
```

#### 6.2.3.2 Fetching Figures/Backgrounds from the eShadow Server

Whenever the app is connected to the main server or a refresh is initiated, it attempts to retrieve the data of the figures and backgrounds stored within the platform and display them on their respective pages. This enables the user to view and load the corresponding object into the platform, allowing them to interact with it using the controller functionality. Here specifically the snippet shown is from the Figure's page; however, similar logic applies to the backgrounds as well. The `_FetchEshadowFigures` method retrieves puppet data from the eShadow server using HTTP GET requests, then decodes and stores the data as an instance of a `TemporaryPuppet` class that is added in an internal list keeping those figure data.

```
Future<void> _FetchEshadowFigures() async {
  setState(() {
    isLoading = true;
  });
  final url = '${global_variables.qrData}/figures';
  try {
    final response = await http.get(Uri.parse(url));
    if (response.statusCode == 200) {
      final List<dynamic> data = json.decode(response.body);
```

```
      final List<TemporaryPuppet> loadedPuppets = data.map((item) {
        return TemporaryPuppet(
          id: item['id'],
          jsonFile: item['jsonFile'],
          name: item['name'],
          thumbnail: base64Decode(item['thumbnail']),
        );
      }).toList();
      setState(() {
        temporary_puppets = loadedPuppets;
        global_variables.isFigureDataLoaded = true;
        global_variables.puppetsData = loadedPuppets;
        isLoading = false;
      });
    } else {
      setState(() {
        isLoading = false;
      });
    }
  } catch (e) {
    setState(() {
      isLoading = false;
    });
  }
}
```

#### 6.2.3.3 Loading Figures to the Stage

The user can load a figure or background into the scene and manipulate them. This is achieved by using the RESTful API embedded within eShadow. By receiving the ID of the desired object and the pre-allocated ID distributed to the user by the system, eShadow loads the pointed-at object into the scene and creates an association with it using a unique ID that allows them to be used to send movement, resize, or other similar commands. Here specifically the snippet shown is from the Figure's page; however, similar logic applies to the backgrounds as well. The `_loadToStage` method sends a request to the server to load a puppet to the stage, identified by its ID. Upon success, the user is navigated to the `PuppetController` screen.

```
Future<void> _loadToStage(int index) async {
  String figureId = temporary_puppets[index].id;
  String url = '${global_variables.qrData}/load_figure_by_id/$figureId/
      puppet_id=${global_variables.puppetID}';
  try {
    final response = await http.get(Uri.parse(url));
    if (response.statusCode == 200) {
      _navigateToControllerScreen(context);
    }
```

```
  } catch (e) {
    ...
  }
}
```

#### 6.2.3.4 Art Style Change Functionality

By connecting to the style transform server, the user can long-press on an image and send it to the mentioned server. The server then returns a new image saved as a new entry in the image gallery. The `_ArtstyleChange` method is responsible for sending the image to the style transform server for art style transformation, and saving the resulting image. The image is sized down before sending to reduce processing time.

```
Future<void> _ArtstyleChange(int index) async {
  if (global_variables.aiServerAddress == "no data yet") {
    return;
  }

  File originalImage = photos[index].mainImage;
  Uint8List imageBytes = await originalImage.readAsBytes();

  // Decode image dimensions
  ui.Image decodedImage = await decodeImageFromList(imageBytes);
  int originalWidth = decodedImage.width;
  int originalHeight = decodedImage.height;

  // Resize the image if necessary
  int targetWidth = originalWidth > 512 ? 512 : originalWidth;
  int targetHeight = originalHeight > 512
      ? (originalHeight * 512 / originalWidth).round()
      : originalHeight;

  Uint8List resizedImageBytes = await FlutterImageCompress.
      compressWithList(
    imageBytes,
    minWidth: targetWidth,
    minHeight: targetHeight,
    quality: 90,
    format: CompressFormat.png,
  );

  // Send the resized image to the AI server
  var request = http.MultipartRequest(
    'POST',
    Uri.parse('${global_variables.aiServerAddress}/change_artstyle'),
  );
```

```
  request.files.add(
    await http.MultipartFile.fromPath('image', resizedImageFile.path,
      contentType: MediaType('image', 'png'),
    ),
  );

  var streamedResponse = await request.send();
  if (streamedResponse.statusCode == 200) {
    Uint8List responseBytes = await streamedResponse.stream.toBytes();

    // Save the generated image
    final Directory appDir = await getApplicationDocumentsDirectory();
    final String imagesDirPath = '${appDir.path}/images';
    File newImageFile = await File('$imagesDirPath/generated_image.png')
        .writeAsBytes(responseBytes);

    // Add to photo list
    setState(() {
      photos.add(Photo(newImageFile, newThumbnailFile));
    });
  }
}
```

#### 6.2.3.5 Creating a Drawn Parts Figure

The application offers the ability to create figures and backgrounds. In the figure creation process, there are two options: ‘creating a figure from a real object or person’ and ‘creating a figure from drawn parts’.

For the ‘creating a figure from drawn parts’ option, the specific code from that section will be used to describe the main logic. This is because it is slightly simpler compared to ‘creating a figure from a real object or person’, while still encompassing all the essential components.

Background creation follows a similar pattern and is even simpler, so it will not be referenced separately.

In the ”creating a figure from drawn parts” process, the application utilizes several image processing steps to transform an uploaded or captured image into a figure. The steps include resizing the image, removing the background, performing connected component analysis for segmentation, and allowing the user to manipulate and assemble the parts into a complete figure.

#### 6.2.3.6 Resizing and Background Removal

After Initiating the process, the application resizes it to fit predetermined dimensions, ensuring optimal processing. The resized image is then sent to the main server for background removal. The server returns a modified image with the background eliminated, facilitating the next step of segmentation.

```
Future<void> _resizeImage() async {
// Resize image
  img_lib.Image? originalImage = img_lib.decodeImage(imageBytes);
  img_lib.Image resizedImage = img_lib.copyResize(
    originalImage!,
    width: newWidth,
    height: newHeight,
  );
  Uint8List resizedImageBytes = Uint8List.fromList(img_lib.encodeJpg(
      resizedImage));
  local_variables.croppedPhoto = resizedImageBytes;
}
// Send the resized image to the server for background removal
Future<void> _removeBackground() async {
  var request = http.MultipartRequest(
    'POST', Uri.parse('${global_variables.qrData}/remove_bg'));
  request.files.add(
    await http.MultipartFile.fromPath('image', tempFile.path));
  var streamedResponse = await request.send();
  if (streamedResponse.statusCode == 200) {
    bgRemovedImage = Uint8List.fromList(await streamedResponse.stream.
        toBytes());
    local_variables.backgroundRemovedPhoto = bgRemovedImage;
  }
}
```

#### 6.2.3.7 Part Segmentation

The background-removed image is sent back to the main server and is analyzed for connected components using the alpha channel to segment it into distinct parts. These parts are cropped into separate images, packaged and sent back to the application.

```
Future<void> _segmentParts() async {
  var request = http.MultipartRequest(
    'POST', Uri.parse('${global_variables.qrData}/segment-parts'));
  request.files.add(await http.MultipartFile.fromPath('image', tempFile.
      path));
  var response = await request.send();
  if (response.statusCode == 200) {
    var bytes = await response.stream.toBytes();
```

```
    var filePath = await _saveZip(bytes, 'downloaded.zip');
    await unzipFile(filePath);
  }
}
```

#### 6.2.3.8 Assembling the Puppet from Parts

After segmenting the figure into distinct parts, the application provides an interactive assembly interface. Users can rearrange and connect these parts into a coherent figure. This interactivity relies on maintaining positions, transformations, and joint information within shared variables and handling user gestures effectively.

#### 6.2.3.9 Data Structures and Variables

A set of global variables keeps track of the state of the assembly. These are defined in a shared variables file:

```
...
...
...
Uint8List? backgroundRemovedPhoto;
List<Uint8List> puppetParts = [];
...
List<Joint> joints = [];
int controllerPartIndex = -1;
...
...
List<Size> imageSizes = [];
Map<int, Uint8List> imagePixelData = {};
List<Offset> positions = [];
...
...
...
```

Here, `puppetParts` holds the segmented figure components, `positions` stores the on-screen coordinates of each part. The `joints` list stores stores the on-screen coordinates of associations (or ”connections”) between parts. Due to design decisions, it was decided to have only one part that the user can grab to move the figure. To further simplify the creation process, it was defined that the control part is the part at the highest position on the screen. This decision was made after observing testers placing the part they want to control almost always at the highest position. Additionally, hints have been incorporated to guide the user. The index of the respective control part is stored in the `controllerPartIndex` variable.

#### 6.2.3.10 Interactivity and Part Manipulation

The assembly interface presents each puppet part as a movable widget. This is achieved using Flutter's `GestureDetector` and transformation matrices. Users can drag individual parts and put them in the desired position.The code snippet below demonstrates how parts are rendered and how user gestures are handled to transform them:

```
return Positioned(
  left: positions[index].dx,
  top: positions[index].dy,
  child: GestureDetector(
    onScaleStart: (details) {
      // Bring the tapped part to the front
      bringToFront(index);
      activeIndex = index;

      // Store initial transform states
      _initialFocalPoint = details.focalPoint;
      _initialPosition = positions[index];
    },
    onScaleUpdate: (details) {
      // Update position to maintain the part under the user's finger
      final offset = details.focalPoint - _initialFocalPoint;
      positions[activeIndex] = _initialPosition + offset;
    },
    child: Image.memory(puppetParts[index]),
  ),
);
```

This approach ensures intuitive manipulation: each part can be individually placed and oriented in the user's desired configuration, with changes immediately reflected on-screen.

#### 6.2.3.11 Joint Creation and Part Linkage

By moving the desired parts around, the user can connect them at desired positions and create 'joints' between them. Joints are an abstraction of actual figure joints employed in traditional Greek shadow theater allow parts to be connected so that one can influence the other's position and orientation. When a user long-presses overlapping parts, the system determines which parts are present at that point using pixel-level transparency checks. Two overlapping parts can be "joined," restricting their independent movement and visually tying them together.

```
void _handleLongPress(Offset globalPosition) {
  // Identify overlapping parts at the press location
  List<int> overlappingIndices = _findOverlappingParts(globalPosition);

  if (overlappingIndices.length >= 2) {
```

```
    // Create a joint between the top two overlapping parts
    int index1 = overlappingIndices[0];
    int index2 = overlappingIndices[1];

    Offset localPosition1 = _getLocalPosition(globalPosition, index1);
    Offset localPosition2 = _getLocalPosition(globalPosition, index2);

    Joint joint = Joint(
      partIndex1: index1,
      partIndex2: index2,
      positionInImage1: localPosition1,
      positionInImage2: localPosition2,
    );

    setState(() {
      joints.add(joint);
      // Once joined, parts may become non-movable depending on the logic
      isPartMovable[index1] = false;
      isPartMovable[index2] = false;
    });
  }
}
```

By introducing these joints, users can design articulated figures with complex relationships among components. For example, a character's arm can be "attached" to a torso so that the arm moves in tandem with the torso rather than independently.

#### 6.2.3.12 Making Created Figure Data eShadow Compatible

After defining the figure parts and joints, appropriate transformations are applied to convert screen coordinates into image coordinates. Subsequently, the data needs to be compiled into a format readable by eShadow. eShadow takes a single image with all the parts combined, as well as assembling instructions for defining how to crop the parts and how to join them. Those instructions are defined in a JSON file included with the image. The specific way those are constructed is described thoroughly in Marios Christoulakis's master thesis [7] so it will be skipped here. The `_generateCombinedImageAndJSON` function in Flutter is responsible for creating a combined sprite sheet from individual puppet parts, adjusting the joint positions for accuracy, and generating a JSON file. This JSON representation encodes information about the puppet's structure, including part metadata, joint connections, and positions. The function also handles saving the combined image localy and the JSON file for further use. An overview of the steps in the compilation is presented below.

#### 6.2.3.13 Step 1: Identify Included Parts

The function identifies which parts of the puppet should be included by iterating over all joints. Parts not connected to the main figure are to be ignored. Each joint connects two parts, and their indices are collected. If no joints exist and only one part is present, it defaults to including that part.

```
Set<int> includedPartIndices = {};
for (var joint in local_variables.joints) {
  includedPartIndices.add(joint.partIndex1);
  includedPartIndices.add(joint.partIndex2);
}
if (includedPartIndices.isEmpty && local_variables.puppetParts.length ==
    1) {
  includedPartIndices.add(0); // Default to the first part if no joints
      exist.
}
if (includedPartIndices.isEmpty) {
  return; // Abort if no parts are included.
}
```

#### 6.2.3.14 Step 2: Build Figure Parts

The function constructs a list of `FigurePart` objects. Each object includes metadata such as the part's name, size, index, and whether it is controllable. The names are assigned sequentially (e.g., "Part1", "Part2").

```
int controllerPartIndex = local_variables.controllerPartIndex;
List<FigurePart> figureParts = [];
int partNumber = 1;

for (int index in includedPartIndices) {
  FigurePart part = FigurePart(
    partName: "Part$partNumber", // Assign a name like "Part1", "Part2",
        etc.
    index: index, // Index of the part.
    image: local_variables.puppetParts[index], // Image data for the part
        .
    size: local_variables.imageSizes[index], // Dimensions of the part.
    isControlled: (index == controllerPartIndex), // Indicates if this
        part is controlled.
  );
  figureParts.add(part);
  partNumber++;
}
```

#### 6.2.3.15 Step 3: Arrange Parts Side by Side

The function arranges the parts side by side in a single canvas, calculating the total width and maximum height of the resulting combined image. This ensures that all parts fit properly in the sprite sheet.

```
double totalWidth = 0;
double maxHeight = 0;

for (FigurePart part in figureParts) {
  totalWidth += part.size.width;
  if (part.size.height > maxHeight) {
    maxHeight = part.size.height;
  }
}
```

#### 6.2.3.16 Step 4: Generate Combined Image

The function creates a canvas and draws each part onto it in sequence. The positions of the parts within the combined image are tracked for accurate joint placement.

```
ui.PictureRecorder recorder = ui.PictureRecorder();
Canvas canvas = Canvas(recorder);

double currentX = 0;
Map<int, Offset> partPositionsInImage = {};

for (FigurePart part in figureParts) {
  ui.Image image = await decodeImageFromList(part.image);
  canvas.drawImage(image, Offset(currentX, 0), Paint());
  part.startPoint = Offset(currentX, 0); // Store position in combined
      image.
  partPositionsInImage[part.index] = part.startPoint;
  currentX += part.size.width;
}

ui.Picture picture = recorder.endRecording();
ui.Image combinedImage = await picture.toImage(totalWidth.ceil(),
    maxHeight.ceil());
```

#### 6.2.3.17 Step 5: Adjust Joint Positions

The joint positions are recalculated based on the new positions of the parts in the combined image. This ensures that the joints remain aligned with their respective parts.

```
List<AdjustedJoint> adjustedJoints = [];
for (var joint in local_variables.joints) {
```

```
  if (includedPartIndices.contains(joint.partIndex1) &&
      includedPartIndices.contains(joint.partIndex2)) {
    FigurePart part1 = figureParts.firstWhere((part) => part.index ==
        joint.partIndex1);
    FigurePart part2 = figureParts.firstWhere((part) => part.index ==
        joint.partIndex2);

    Offset part1PositionInImage = partPositionsInImage[part1.index]!;
    Offset part2PositionInImage = partPositionsInImage[part2.index]!;

    AdjustedJoint adjustedJoint = AdjustedJoint(
      figurePart1: part1.partName,
      figurePart2: part2.partName,
      figurePart1Anchor: part1PositionInImage + joint.positionInImage1,
      figurePart2Anchor: part2PositionInImage + joint.positionInImage2,
    );
    adjustedJoints.add(adjustedJoint);
  }
}
```

#### 6.2.3.18 Step 6: Generate JSON Data

The JSON data is generated to represent the structure of the puppet, including part metadata and joint information. This JSON is saved locally and can also be sent to a server.

```
var uuid = Uuid();
String puppetId = uuid.v4();

Map<String, dynamic> jsonData = {
  "figureName": puppetId,
  "figureSpriteSheet": "${puppetId}.png",
  "figurePackPath": "MobileCreatedFigures",
  "pixelsPerUnit": 600,
  "figurePartList": figureParts.map((part) => {
    "partName": part.partName,
    "partStartPoint": {"x": part.startPoint.dx, "y": part.startPoint.dy},
    "partSize": {"x": part.size.width, "y": part.size.height},
    "isControlled": part.isControlled,
  }).toList(),
  "jointList": adjustedJoints.map((joint) => {
    "figurePart1": joint.figurePart1,
    "figurePart2": joint.figurePart2,
    "figurePart1Anchor": {"x": joint.figurePart1Anchor.dx, "y": joint.
        figurePart1Anchor.dy},
    "figurePart2Anchor": {"x": joint.figurePart2Anchor.dx, "y": joint.
        figurePart2Anchor.dy},
  }).toList(),
```

```
};

String jsonString = jsonEncode(jsonData);
```

#### 6.2.3.19 Step 7: Store Image and JSON Localy

Finally, the combined image and JSON file are saved locally by creating an image of the `Puppet` class.

```
File? puppetImage = await saveImage(imageBytes, figureSpriteSheet);
File? thumbnailImage = await resizeImage(
    local_variables.backgroundRemovedPhoto!, thumbnailSpriteSheet);

Puppet puppet = Puppet(
    id: puppetId,
    name: figureName,
    jsonFile: jsonString,
    image: puppetImage!,
    thumbnail: thumbnailImage);

global_variables.localPuppets.add(puppet);
savePuppets();
```

### 6.2.4 Output

The `_generateCombinedImageAndJSON` function integrates multiple components to create a functional puppet object. It ensures that all parts are aligned correctly and generates a JSON file compatible with eShadow. An output sample is displayed below.

```
{
  "figureName": "5f2818b1-7998-4420-beb3-e47e38d16f0d",
  "figureSpriteSheet": "5f2818b1-7998-4420-beb3-e47e38d16f0d.png",
  "figurePackPath": "MobileCreatedFigures",
  "pixelsPerUnit": 600,
  "figurePartList": [
    {
      "partName": "Part1",
      "partStartPoint": {"x": 0.0, "y": 0.0},
      "partSize": {"x": 554.0, "y": 516.0},
      "isControlled": false
    },
    {
      "partName": "Part2",
      "partStartPoint": {"x": 554.0, "y": 0.0},
      "partSize": {"x": 834.0, "y": 821.0},
      "isControlled": true
```

```
    },
    {
      "partName": "Part3",
      "partStartPoint": {"x": 1388.0, "y": 0.0},
      "partSize": {"x": 361.0, "y": 636.0},
      "isControlled": false
    },
    {
      "partName": "Part4",
      "partStartPoint": {"x": 1749.0, "y": 0.0},
      "partSize": {"x": 321.0, "y": 648.0},
      "isControlled": false
    }
  ],
  "jointList": [
    {
      "figurePart1": "Part1",
      "figurePart2": "Part2",
      "figurePart1Anchor": {"x": 291.7247687564234, "y":
          48.88823227132548},
      "figurePart2Anchor": {"x": 1027.5238095238096, "y":
          744.8371873929428}
    },
    {
      "figurePart1": "Part3",
      "figurePart2": "Part1",
      "figurePart1Anchor": {"x": 1481.3720966084275, "y":
          70.87115450496754},
      "figurePart2Anchor": {"x": 361.2247687564234, "y":
          482.1048989379924}
    },
    {
      "figurePart1": "Part4",
      "figurePart2": "Part1",
      "figurePart1Anchor": {"x": 1968.6389003083248, "y":
          80.25111339499836},
      "figurePart2Anchor": {"x": 148.09143542308993, "y":
          491.3715656046588}
    }
  ]
}
```

Figure 6.2: Single Output Image With all the Parts Combined

#### 6.2.4.1 Uploading Local Puppets to the Server

After the newly created figure is instantiated as a `Puppet` class object, it is stored in a list called `local_puppets[]` that keeps all the locally created puppets together. The local puppets are displayed in the puppets tab and can be selected by the user to be loaded to the eShadow Scene.

The `_uploadPuppet` method sends a local puppet's data to the server via an HTTP POST request to be instantiated into the eShadow scene. Just like the loading of a puppet already inside eShadow, the unique puppet ID is sent, with key difference that this time the JSON, and image file are included. The image is encoded into a `Base64` string format in order to be sent.

```
Future<void> _uploadPuppet(int index) async {
  Puppet puppet = local_puppets[index];
  String? puppetName = await _promptForPuppetName();
  if (puppetName == null || puppetName.isEmpty) return;

  String jsonContent = puppet.jsonFile;
  String imageBase64 = base64Encode(await puppet.image.readAsBytes());
  String thumbnailBase64 = base64Encode(await puppet.thumbnail.
      readAsBytes());

  String url = '${global_variables.qrData}/upload_puppet';
  Map<String, dynamic> data = {
    'figureId': puppet.id,
    'puppetName': puppetName,
    'puppetJsonFile': jsonContent,
    'PuppetImage': imageBase64,
    'PuppetThumbnail': thumbnailBase64,
  };

```

```
  try {
    final response = await http.post(
      Uri.parse(url),
      headers: {'Content-Type': 'application/json'},
      body: jsonEncode(data),
    );
    if (response.statusCode == 200) {
      _FetchEshadowFigures();
    }
  } catch (e) {
    ...
  }
}
```

In a similar way, the user can choose to upload figures and background to be stored into the plaform, so that they can be accessed by other users.

#### 6.2.4.2 Saving and Loading Local Puppets

The application saves local puppets using `SharedPreferences` and loads them into memory at startup.

```
Future<void> savePuppets() async {
  SharedPreferences storedPuppetFiles = await SharedPreferences.
      getInstance();
  List<String> ids = global_variables.localPuppets.map((puppet) => puppet
      .id).toList();
  List<String> names = global_variables.localPuppets.map((puppet) =>
      puppet.name).toList();
  List<String> jsonFiles = global_variables.localPuppets.map((puppet) =>
      puppet.jsonFile).toList();
  List<String> puppet_main_images = global_variables.localPuppets.map((
      puppet) => puppet.image.path).toList();
  List<String> puppet_thumbnails = global_variables.localPuppets.map((
      puppet) => puppet.thumbnail.path).toList();

  storedPuppetFiles.setStringList('puppet_ids', ids);
  storedPuppetFiles.setStringList('puppet_names', names);
  storedPuppetFiles.setStringList('puppet_json', jsonFiles);
  storedPuppetFiles.setStringList('puppet_main_images',
      puppet_main_images);
  storedPuppetFiles.setStringList('puppet_thumbnails', puppet_thumbnails)
      ;
}
```

#### 6.2.4.3 Real Time Control

When a figure or background is loaded into the eShadow scene, an app screen opens that enables the user to manipulate in real time the instantiated object using touch gestures. Since the interfaces for figure control and background control are pretty similar, snippets only from the figure controller a goint to be presented from now on.

#### 6.2.4.4 Initialization and UDP Setup

On initialization, the widget creates a UDP socket and retrieves the server address from a globally stored URL to be used to send messages. The server port is predefined.

```
late RawDatagramSocket udpSocket;
late String flaskServerIp;
final int flaskServerPort = 5005; // UDP port on the server

@override
void initState() {
  super.initState();
  _initializeUdpSocket();

  // Extract host (IP/domain) from the global variables
  Uri parsedUri = Uri.parse(global_variables.qrData);
  flaskServerIp = parsedUri.host;
}

void _initializeUdpSocket() async {
  udpSocket = await RawDatagramSocket.bind(InternetAddress.anyIPv4, 0);
  udpSocket.listen((event) {
    if (event == RawSocketEvent.read) {
      Datagram? datagram = udpSocket.receive();
      if (datagram != null) {
        print('Received: ${String.fromCharCodes(datagram.data)}');
      }
    }
  });
}
```

#### 6.2.4.5 Handling Gestures

The `GestureDetector` widget is responsible for interpreting user input as gestures. It differentiates between single-finger and two-finger interactions, enabling both precise movements and more complex transformations like rotation.

```
// GestureDetector to handle user interactions
GestureDetector(
```

```
  onScaleStart: (details) {
    if (details.pointerCount == 2) {
      // Two-finger rotation
      TurnFigure(details.localFocalPoint);
    } else {
      // Single-finger drag begins
      sendTouchDown(details.localFocalPoint);
      sendTimer = Timer.periodic(
        Duration(milliseconds: sendInterval),
        (timer) => _sendDataAtIntervals()
      );
    }
  },
  onScaleUpdate: (details) {
    // Update position during single-finger movement
    if (details.pointerCount == 1) {
      sendTouchData(details.localFocalPoint);
    }
  },
  onScaleEnd: (details) {
    // Finger lifted, send touch-up and stop updates
    sendTouchUp();
    sendTimer?.cancel();
    sendTimer = null;
  },
  child: Container(
    // UI elements, instructions, and slider
  ),
);
```

#### 6.2.4.6 Capturing Movement

Beyond simply recognizing gestures, the system must relay the puppet's movements to the server in real time. To achieve this, the application periodically transmits position updates at a predefined interval. These updates ensure smooth, continuous motion on the server side without causing network congestion.

The chosen update interval (`sendInterval`) was determined through trial and error, balancing responsiveness against network load. Threshold values further refine the data, filtering out negligible motion and clamping excessively large movements for a stable and predictable user experience.

```
// Variables for tracking position and sending intervals
Offset? lastSentPosition;
Offset? currentPosition;
Timer? sendTimer;
final int sendInterval = 5; // Interval in milliseconds
```

```
6
7  void sendTouchData(Offset position) {
8    currentPosition = position;
9  }
10
11 void _sendDataAtIntervals() {
12   if (currentPosition != null && (lastSentPosition == null ||
         currentPosition != lastSentPosition)) {
13     if (lastSentPosition == null) {
14       lastSentPosition = currentPosition;
15     }
16
17     // Compute movement differences and apply threshold adjustments
18     Offset differentialPosition = (currentPosition! - lastSentPosition!)
           / divider;
19     double adjustedDx = _applyThreshold(differentialPosition.dx);
20     double adjustedDy = _applyThreshold(differentialPosition.dy);
21
22     Map<String, dynamic> puppetData = {
23       "event": "onPanUpdate",
24       "userID": global_variables.puppetID,
25       "pos_x": adjustedDx,
26       "pos_y": -adjustedDy,
27     };
28
29     // Send updated position to the server
30     udpSocket.send(
31       utf8.encode(jsonEncode(puppetData)),
32       InternetAddress(flaskServerIp),
33       flaskServerPort
34     );
35
36     lastSentPosition = currentPosition;
37   }
38 }
```

#### 6.2.4.7 Control Messages

Depending on the situation, there are a few JSON-encoded UDP control messages utilized to facilitate for real time communication between the controller and the eShadow Platform. Those control messages mainly contain a command called 'event' and the unique ID associated with the object being manipulated. For reference, the commands used by the figure controller are presented below.

- **sliderChanged:** Triggered when the user adjusts the size slider. This message carries a new `scale` value and updates the figure's size on the Scene.

- **onTouchDown:** Sent when the user first touches the screen to begin interacting with the puppet. In the eShadow platform this is the equivalent to a computer mouse click down and un-pins the control part, making it movable. It includes the initial `pos_x` and `pos_y` coordinates of the touch.
- **onTouchUp:** Dispatched when the user lifts their finger, signaling the end of an interaction. In the eShadow platform this is the equivalent to a computer mouse click up and pins down the control part, stabilizing the figure.
- **onPanUpdate:** Emitted periodically while the user drags the puppet. It contains `pos_x` and `pos_y` offsets relative to the last known position.
- **Turn:** Invoked by a two-finger gesture. It provides the coordinates for rotation events, enabling the puppet's orientation to change.

## 6.3 Backend (Servers)

### 6.3.1 Main Server

- Built using Flask, it exposes RESTful API endpoints and handles HTTP and UDP-based communication for the system. It integrates third-party libraries such as OpenCV, Rembg, and ONNX Runtime for advanced image processing and segmentation tasks.
- **Key functionalities:**
  - **Image Processing and Background Removal:** Exposes the endpoint `POST /remove_bg` for background removal using the `rembg` [13] library.
  - **Image Segmentation:** Exposes the endpoint `POST /segment-parts` for processing RGBA images to recognize and crop all the parts separated by the transparent layer left from background removal. This is to automatically separate drawn parts of a figure in a piece of paper or any other surface.
  - **Relay Data to Unity:** Relays data to the Unity platform through:
    - HTTP endpoints like `/figures` and `/upload_puppet`.
    - Low-latency control messages sent via UDP on port `8080`.
  - **Heartbeat and Status Monitoring:** Provides a `GET /heartbeat` endpoint to verify server status.
  - **Networking and Configuration:**
    - Listens for REST API requests on port `5000`.
    - Receives UDP messages from the Mobile Application on port `5005`.

    - Relays UDP messages to Unity's port `8080`.

- **Deployment:** The Main Server is packaged as a standalone executable using `PyInstaller` for easy end-user distribution.

- **Technologies and Libraries:**

  - **Flask:** Provides RESTful API endpoints [46].
  - **Rembg:** Used for background removal tasks [13].
  - **OpenCV:** Enables image segmentation and manipulation [17].
  - **ONNX Runtime:** Facilitates efficient model inference for advanced tasks [12].
  - **Pillow:** Handles image processing tasks such as format conversion [48].

#### 6.3.1.1 Background Removal

The Main Server provides advanced image processing capabilities, including background removal, powered by the `rembg` [13] library. The following code snippet shows the implementation of the endpoint `POST /remove_bg`, which processes uploaded images and returns the background-removed output:

```
@app.route('/remove_bg', methods=['POST'])
def process_image():
    if 'image' not in request.files:
        return "No image uploaded", 400

    file = request.files['image']
    try:
        image = Image.open(file)  # Load the uploaded image
        rm_bg_img = remove(image, session=session)  # Remove background

        # Save the processed image to memory and return it
        img_io = BytesIO()
        rm_bg_img.save(img_io, format='PNG')
        img_io.seek(0)
        return send_file(img_io, mimetype='image/png', as_attachment=True
            , download_name='processed_image.png')
    except Exception as e:
        return jsonify({'error': str(e)}), 500
```

#### 6.3.1.2 Image Segmentation for Puppet Parts

The Main Server implements an endpoint, `POST /segment-parts`, to segment puppet parts from RGBA images. The segmentation uses connected components analysis and outputs the segmented parts as a ZIP file. The key implementation is shown below:

```python
@app.route('/segment-parts', methods=['POST'])
def process_puppet_image():
    if 'image' not in request.files:
        return "No image uploaded", 400

    file = request.files['image']
    try:
        # Open and process the image
        image = Image.open(file).convert('RGBA')
        output_files = segment_image(image)  # Segment the image into
            parts

        # Create a ZIP file in memory with segmented parts
        zip_buffer = io.BytesIO()
        with ZipFile(zip_buffer, 'w') as zip_file:
            for i, img_bytes in enumerate(output_files):
                img_name = f"segment_{i + 1}.png"
                zip_file.writestr(img_name, img_bytes)

        zip_buffer.seek(0)  # Reset the buffer before sending
        return send_file(zip_buffer, mimetype='application/zip',
            as_attachment=True, download_name='segments.zip')
    except Exception as e:
        return jsonify({'error': str(e)}), 500
```

#### 6.3.1.3 How the `segment_image` Function Works

The `segment_image` function is a key component of the Main Server, designed to segment an RGBA image into distinct parts. This segmentation process is fundamental to puppet creation, as it allows individual components to be extracted and manipulated separately. The function operates as follows:

1. **Input Conversion:** The function accepts an image in RGBA format, which is then converted into a NumPy array for efficient processing with OpenCV. The function ensures that the input includes an alpha channel (transparency), as segmentation relies on identifying non-transparent regions.

2. **Alpha Channel and Mask Generation:** The alpha channel is extracted from the image to create a binary mask, where non-transparent pixels are marked as foreground. Connected component analysis is applied to this mask to identify distinct regions.

3. **Component Segmentation:** Each connected component is processed independently. A mask for the current component is applied to the image, isolating the correspond-

ing region. The image is cropped to the bounding box of the component to eliminate unnecessary background.

4. **Filtering by Region Size:** To ensure only significant components are retained, the segmented regions are filtered based on their area. Components smaller than a threshold (10% of the largest region) are discarded.

5. **Encoding and Output:** Each retained region is encoded as a PNG image and stored in memory. The function returns a list of these encoded images, which can be saved or packaged for further use.

The segmentation process ensures that only meaningful regions of the image are preserved, allowing users to interact with distinct puppet parts. This functionality plays a vital role in the overall workflow of the eShadow platform.

```
def segment_image(image):
    # Convert the image to a numpy array
    image_np = np.array(image)

    # Ensure the image includes an alpha channel
    if image_np.shape[2] != 4:
        raise ValueError("Image does not have an alpha channel.")

    # Convert RGBA to BGRA for OpenCV compatibility
    image_np = image_np[:, :, [2, 1, 0, 3]]

    # Extract the alpha channel and create a binary mask
    alpha_channel = image_np[:, :, 3]
    mask = alpha_channel != 0
    num_labels, labels = cv2.connectedComponents(mask.astype(np.uint8))

    output_files = []
    areas = []  # Store areas of ROIs for filtering

    # Iterate over connected components to segment the image
    for label in range(1, num_labels):
        component_mask = (labels == label).astype(np.uint8)

        # Create a mask for the current component
        segmented_image_np = image_np.copy()
        for c in range(3):  # Apply mask to each color channel
            segmented_image_np[:, :, c] = segmented_image_np[:, :, c] *
                component_mask
        segmented_image_np[:, :, 3] = segmented_image_np[:, :, 3] *
            component_mask

        # Find the bounding box of the component
```

```
        contours, _ = cv2.findContours(component_mask, cv2.RETR_EXTERNAL,
            cv2.CHAIN_APPROX_SIMPLE)
        for cnt in contours:
            x, y, w, h = cv2.boundingRect(cnt)
            ROI = segmented_image_np[y:y + h, x:x + w]

            # Further crop the ROI to non-transparent pixels
            alpha = ROI[:, :, 3]
            ys, xs = np.nonzero(alpha)
            if len(xs) > 0 and len(ys) > 0:
                min_x, max_x = xs.min(), xs.max()
                min_y, max_y = ys.min(), ys.max()
                ROI_cropped = ROI[min_y:max_y + 1, min_x:max_x + 1]
                area = ROI_cropped.shape[0] * ROI_cropped.shape[1]
                areas.append((area, ROI_cropped))

    if not areas:
        return []

    # Filter ROIs by size and encode as PNG
    max_area = max(areas, key=lambda x: x[0])[0]
    for area, ROI in areas:
        if area >= 0.1 * max_area:  # Retain only significant ROIs
            _, encoded_image = cv2.imencode('.png', ROI)
            output_files.append(encoded_image.tobytes())

    return output_files
```

#### 6.3.1.4 Heartbeat and Status Monitoring

To ensure the availability and status of the Main Server, a simple heartbeat endpoint is implemented, `GET /heartbeat`, this endpoint is used both by eShadow and the mobile application to verify the server's online status. This endpoint responds with the server's status:

```
@app.route('/heartbeat', methods=['GET'])
def heartbeat():
    return jsonify({"status": "online"}), 200
```

### 6.3.2 Style Transform Server

The Style Transform Server is a modular component of the eShadow LCE system, meaning that the system is designed to function even in its absence. This design decision will be further discussed in a later section. This component's main goal is to apply traditional shadow theater art styles to user-uploaded images without the need for some kind of prompt explaining how to do the style transform, while at the same time preserving key characteristics of the input image

through the use of advanced Diffusion [51] based models. Secondary goal for this server is to facilitate the artistic transformation process while maintaining high responsiveness to meet user demands.

The server employs state-of-the-art AI models, including Stable Diffusion [51] and ControlNet [68], optimized with GPU acceleration for efficient processing. The pipeline leverages a locally trained LoRA [25] model to apply highly specific stylistic transformations tailored to traditional Greek shadow theater aesthetics. By offloading computationally intensive tasks to the GPU and using techniques like ‘float16‘ precision and model CPU offloading along with various optimizations the server attempts to be locally deployable even at computer with mid-range gpu hardware.

Key functionalities of the Style Transform Server include:

- **Art Style Transformation:** Utilizes a combination of Stable Diffusion and ControlNet’s `softedge` model to transform user-uploaded images into traditional shadow theater aesthetics while retaining the original image’s key characteristics.
- **Endpoint-Based Communication:** Exposes RESTful endpoints, such as `POST /change_artstyle`, for interation with other components.
- **QR Code Integration:** Dynamically generates QR codes containing the server’s connection details, allowing for is in a local environment.

#### 6.3.2.1 Model Initialization

The following code snippet illustrates the initialization process for the AI models used in the Style Transform Server. It defines the paths for the required models, checks hardware availability (CPU or GPU), and loads the necessary components, including ControlNet, Stable Diffusion, and a locally trained LoRA model. Key optimizations, such as using `float16` for faster computations and enabling CPU offloading, are employed to improve efficiency.

```
# Model paths
lora_path = "models/LORA/karagiozis.safetensors" #Locally Trained LoRA
model_path = "models/Stable_Diff/output" #Stable Diffusion Model
softedge_path = "models/ControlNet"  #ControlNet Softedge Model

# Load models into the hardware
def init_models(model_path, softedge_path, lora_path):
    # Check if CUDA is available
    device = "cuda" if torch.cuda.is_available() else "cpu"
    # Load models to device
    controlnet = ControlNetModel.from_pretrained(
        softedge_path, torch_dtype=torch.float16, use_safetensors=True
    )
    pipeline = StableDiffusionControlNetImg2ImgPipeline.from_pretrained(
```

```
        model_path,
        controlnet=controlnet,
        torch_dtype=torch.float16, # float16 improves speed
        use_safetensors=True,
        safety_checker=None,
    ).to(device)

    pipeline.load_lora_weights(lora_path) # Load LoRA weights

    # Noise Scheduler
    pipeline.scheduler = UniPCMultistepScheduler.from_config(pipeline.
        scheduler.config)
    pipeline.enable_model_cpu_offload()
    print("Loaded models to device")
    return pipeline
```

#### 6.3.2.2 Task Queue System for Concurrent Requests

To be able to handle multiple style transformation requests, the server employs a simple queue system.

```
# Task structure
class Task:
    def __init__(self, input_image, callback_event, response_container,
        prompt=""):
        self.input_image = input_image
        self.prompt = prompt
        self.callback_event = callback_event
        self.response_container = response_container

# Task processing worker
def worker():
    while True:
        task = task_queue.get()
        if task is None:
            break  # Graceful shutdown
        try:
            input_image = task.input_image
            prompt = task.prompt
            control_image = get_control_image(input_image)
            with pipe_lock:
                with torch.autocast("cuda"):
                    generated_image = pipe(
                        prompt=prompt,
                        image=input_image,
                        num_inference_steps=8,
                        control_image=control_image
```

```
                    ).images[0]
            output_path = f'generated_images/generated_image_{uuid.uuid4
                ().hex}.png'
            os.makedirs(os.path.dirname(output_path), exist_ok=True)
            generated_image.save(output_path)
            task.response_container['output_path'] = output_path
        except Exception as e:
            task.response_container['error'] = str(e)
        finally:
            task.callback_event.set()
            task_queue.task_done()
```

#### 6.3.2.3 Control Image Generation

ControlNet [68] requires a control image that preserves the structure of the input image while allowing stylistic transformations. The server uses a HED detector to generate the control image, ensuring compatibility with ControlNet. The following function demonstrates the implementation:

```
def get_control_image(init_image):
    processor = HEDdetector.from_pretrained('lllyasviel/Annotators')
    ctrl_image = processor(init_image, safe=True)
    ctrl_image = ctrl_image.resize(init_image.size)
    ctrl_image.save("Images/control_image.png")
    return ctrl_image
```

#### 6.3.2.4 Endpoint for Style Transformation

The server exposes a RESTful endpoint `/change_artstyle` to accept style transformation requests. This endpoint receives the input image, processes the request using the task queue, and returns the generated image to the client. The goal of this component is to be prompt-less so the prompt is set as blank.

```
@app.route('/change_artstyle', methods=['POST'])
def artstyle_changer():
    if 'image' not in request.files:
        return "No image uploaded", 400

    input_image = request.files['image']
    try:
        input_image = Image.open(input_image).convert("RGB")
    except Exception as e:
        return jsonify({'error': 'Invalid image format'}), 400

    prompt = request.form.get('prompt', "")  #the application is to be
        promptless
```

```
13
14      callback_event = threading.Event()
15      response_container = {}
16
17      task = Task(
18          input_image=input_image,
19          callback_event=callback_event,
20          response_container=response_container,
21          prompt=prompt
22      )
23      task_queue.put(task)
24      callback_event.wait()
25
26      if 'error' in response_container:
27          return jsonify({'error': response_container['error']}), 500
28
29      output_path = response_container.get('output_path')
30      if not output_path:
31          return jsonify({'error': 'Unknown␣error␣occurred'}), 500
32
33      return send_file(output_path, mimetype='image/png', download_name='
            generated_image.png')
```

#### 6.3.2.5 QR Code Generation for Connectivity

To allow for local client-server connectivity, the server dynamically generates a QR code containing its connection details. This QR code allows users to connect to the server by scanning it with the mobile application.

```
1  def generate_and_save_qr_code():
2      ip_address = get_local_ip()
3      if not ip_address:
4          print("Could␣not␣determine␣local␣IP␣address.")
5          return
6
7      server_url = f"http://{ip_address}:5110"
8      qr = qrcode.QRCode(version=1, box_size=10, border=4)
9      qr.add_data(server_url)
10     qr.make(fit=True)
11     img = qr.make_image(fill='black', back_color='white')
12     qr_code_path = 'ai_server_qr_code.png'
13     img.save(qr_code_path)
14     print(f"QR␣code␣saved␣to␣{qr_code_path}")
```

## 6.4 eShadow Platform

The eShadow platform is a core component of the overall system, responsible for rendering and managing the digital scene where puppets and backgrounds are visualized and controlled. Expanding upon the original eShadow [7], this version introduces real-time object control and manipulation capabilities. Additionally, it enables the sharing of one's creations through a local puppet-background upload and distribution system. Furthermore, it includes some minor quality-of-life enhancements as well as a scene moderation system.

This section outlines the implementation details of the upgraded eShadow platform, focusing on its core functionalities.

### 6.4.1 Networking Manager: Communication Hub

The `NetworkingManager` serves as the core communication manager in the Unity platform and is implemented based on the `Singleton Design Pattern`. It is responsible for communicating with the main server and receive information and commands. This class integrates HTTP and UDP protocols to handle data exchange in real-time.

**Key Functionalities:**

- **HTTP Listener:** Handles REST API requests for uploading, loading, and managing puppets and backgrounds.
- **UDP Server:** Manages low-latency control messages for real-time puppet and background manipulation.
- **Main Thread Queue:** Ensures thread-safe execution of Unity actions by queuing tasks that interact with the game objects.

The server initialization code is presented below.

```
void Start()
{
    listener = new HttpListener();
    listener.Prefixes.Add(url);
    listener.Start();
    listener.BeginGetContext(OnRequest, listener);

    udpEndPoint = new IPEndPoint(IPAddress.Any, 8080);
    udpServer = new UdpClient(udpEndPoint);
    udpServer.BeginReceive(OnUdpData, udpServer);
}
```

### 6.4.2 Real-Time Puppet Manipulation

Real-time manipulation of puppets is achieved through UDP events, allowing users to control puppet positions, scale, and orientation interactively.

**Key Events:**

- **`onPanUpdate`:** Updates the puppet's position based on user input.
- **`sliderChanged`:** Adjusts the scale of the puppet dynamically.
- **`Turn`:** Flips the puppet horizontally, simulating a "turning" action.

```
case "onPanUpdate":
    QueueOnMainThread(() => MoveFigure(userID, position));
    break;
case "sliderChanged":
    QueueOnMainThread(() => ScaleFigure(userID, scale));
    break;
case "Turn":
    QueueOnMainThread(() => TurnFigure(userID));
    break;
```

```
public void MoveFigure(string userID, Vector2 position)
{
    HingeJoint2D dragJoint = dragJointtDictionary[userID];
    dragJoint.transform.position += new Vector3(position.x, position.y,
        0);
}
```

### 6.4.3 Real-Time Scenery Manipulation

Scenery manipulation supports position updates, scaling, and rotation in real time, enabling users to control background elements dynamically.

**Key Events:**

- **`SceneryOnPanUpdate`:** Updates scenery position and scale.
- **`SceneryOnPinch`:** Adjusts the scale of the scenery using pinch gestures.

```
public void MoveScenery(string userID, Vector2 position, float
    scaleFactor, float rotationAngle)
{
    GameObject scenery = SceneryDictionary[userID];
    scenery.transform.position += new Vector3(position.x, position.y, 0);
```

```
    scenery.transform.localScale = new Vector3(scaleFactor, scaleFactor,
        scenery.transform.localScale.z);
    scenery.transform.Rotate(0, 0, rotationAngle);
}
```

### 6.4.4 Dictionary Management for Puppets and Scenery

Each device is assigned two unique IDs by the server: one for referencing a single figure and the other for a background. These unique IDs are used to associate a pointer to the respective figure or background in a dictionary for efficient access and management.

A device can have only one figure and one background loaded in the scene at any given time. When a new figure or background is loaded, the previously loaded one is destroyed, and the new instance is assigned the respective unique ID. The dictionaries store pointers to the figures and backgrounds, which can be retrieved using their unique IDs, ensuring efficient and organized management of the loaded elements. Below is a snippet o code from the loading of a new figure.

```
// Add instantiated puppet to dictionary
public void AddPuppetToDictionary(string puppetId, GameObject
    puppetObject)
{   // If figure already exists
    if (PuppetDictionary.ContainsKey(puppetId))
    {   // Destroy old and re-assign ID
        Destroy(PuppetDictionary[puppetId]);
        PuppetDictionary[puppetId] = puppetObject;
    }
    else
    {   // Add and assosiation between new figure and ID
        PuppetDictionary.Add(puppetId, puppetObject);
    }
}
```

### 6.4.5 Localy Stored Puppet Loading and Initialization

The `loadPuppet` class is responsible for loading and initializing localy stored puppet figures from predefined configurations stored in JSON files. It dynamically constructs puppet GameObjects with their respective parts, joints, and other properties.

**Key Responsibilities:**

- Reads JSON definitions for puppet parts, joints, and associated assets (e.g., spritesheets).
- Constructs the puppet as a hierarchy of GameObjects, each representing a specific figure part.

- Adds physical properties like rigid bodies and polygon colliders to enable interaction.
- Registers the puppet in the `NetworkingManager` for further manipulation and control.

The following steps are taken in order to construct the figure.

**Step 1 : Loading Figure Data**

```
public IEnumerator LoadFigure(string figureId, string userId, List<
    Vector3> positions = null)
{
    figureDefinition = new ShadowFigureDefinition();
    importedFigurePath = Path.Combine(Application.streamingAssetsPath, "
        Figures");
    figureMenuFilePath = Path.Combine(Application.streamingAssetsPath,
        Constants.MenuItemsJsonFile);

    // Read JSON file for figure definition
    WWW www = new WWW(Constants.FixPath(figureMenuFilePath));
    yield return www;
    string dataJson = www.text;

    menuItems = JsonUtility.FromJson<MenuItemsDefinition>(dataJson);
    var figureToLoad = menuItems.figureMenuItems.Find(x => x.id ==
        figureId);
    ...
```

**Step 2: Constructing Figure Parts** For each part of the figure, a GameObject is created and configured with a sprite renderer, rigid body, and collider. The parts are also assigned tags for controlled and non-controlled components.

```
foreach (var part in figureDef.figurePartList)
{
    // Create sprite for the part
    var spriteRectangle = new Rect(part.partStartPoint.x, tempWWWTexture.
        height - (part.partStartPoint.y + part.partSize.y), part.partSize.
        x, part.partSize.y);
    Sprite figurePartSprite = Sprite.Create(tempWWWTexture,
        spriteRectangle, new Vector2(0f, 1f), figureDef.pixelsPerUnit);

    GameObject figurePart = new GameObject();
    figurePart.name = part.partName;
    SpriteRenderer figurePartSpriteRenderer = figurePart.AddComponent<
        SpriteRenderer>();
    figurePartSpriteRenderer.sprite = figurePartSprite;

    // Assign physical properties
    figurePart.AddComponent<Rigidbody2D>();
```

```
    figurePart.AddComponent<PolygonCollider2D>();
    figurePart.transform.SetParent(figure.transform, false);
}
```

**Step 3: Configuring Joints** The figure's joints are dynamically created based on the configuration in the JSON file. Each joint connects two figure parts and uses a hinge joint for movement simulation.

```
foreach (var joint in figureDef.jointList)
{
    var part1 = GameObject.Find(figure.name + "/" + joint.figurePart1);
    var part2 = GameObject.Find(figure.name + "/" + joint.figurePart2);

    // Skipping detailed joint anchor calculations for brevity.
    // Refer to full source code for joint distance and positioning logic
        .

    var hingeJoint = part1.AddComponent<HingeJoint2D>();
    hingeJoint.connectedBody = part2.GetComponent<Rigidbody2D>();
    hingeJoint.anchor = RevertYAxis(joint.figurePart1Anchor);
    hingeJoint.enableCollision = false;
}
```

**Step 4: Integration with `NetworkingManager`** Once the puppet is loaded, it is registered with the `NetworkingManager` to enable real-time manipulation and interaction.

```
NetworkingManager networkingManager = FindObjectOfType<NetworkingManager
    >();
if (networkingManager != null)
{
    networkingManager.AddPuppetToDictionary(userId, figureGameobject);
}
```

### 6.4.6 External Puppet Loading and Initialization

Apart from loading a locally stored figure or background, the platform can receive an object's data stored in an instance of the mobile application and construct the object on the spot. Below, only code from the figure construction is presented since the background construction is similar, if not simpler.

The `loadExternalPuppet` class is responsible for loading puppet definitions and textures provided as JSON strings and base64-encoded image data provided by the `NetworkingManager`.

**Key Responsibilities:**

- Decodes base64-encoded images and constructs a texture for the puppet.
- Dynamically generates puppet parts as GameObjects, with associated sprites, rigid bodies, and colliders.
- Configures joint connections between puppet parts based on the provided definition.
- Registers the puppet in the `NetworkingManager` for further manipulation.

**Loading an External Puppet Figure** The `LoadFigure` coroutine reads puppet data (JSON and base64-encoded image) passed as method arguments, decodes the image, and constructs a puppet GameObject dynamically.

```
public IEnumerator LoadFigure(string figureId, string userId, string
    InputJsonFile, string InputPuppet, List<Vector3> positions = null)
{
    figureDefinition = new ShadowFigureDefinition();

    // Parse JSON for figure definition
    string dataJson = InputJsonFile;
    var figureDef = JsonUtility.FromJson<ShadowFigureDefinition>(dataJson
        );
    figureDefinition = figureDef;

    // Decode base64 image
    byte[] imageBytes = Convert.FromBase64String(InputPuppet);
    Texture2D tempWWWTexture = new Texture2D(2, 2);
    tempWWWTexture.LoadImage(imageBytes);

    GameObject figure = new GameObject();
    string uniqueID = Guid.NewGuid().ToString();
    figure.name = uniqueID;
    figure.tag = Constants.FigureTag;
    figure.layer = LayerMask.NameToLayer(Constants.PhysicsIgnoreLayer);

    // Skipping intermediate steps for creating figure parts and
        registering them.
    // Refer to full source code for details.
```

### 6.4.7 Storing Externally Stored figures and Backgrounds

The platform also supports uploading user-created figures and background to the server to be shared with everyone in the network. This is handled via HTTP requests, where data is serialized and stored for later use.

**Key Functionalities:**

- **Uploading Puppets:** Processes HTTP requests for puppet data, including JSON configuration and images.
- **Uploading Scenery:** Handles similar functionality for scenery assets.

```
else if (type == "upload_puppet")
{
    try
    {
        using (var reader = new StreamReader(request.InputStream, request
            .ContentEncoding))
        {
            string requestBody = reader.ReadToEnd();
            var requestData = JsonUtility.FromJson<UploadPuppetData>(
                requestBody);
            UploadPuppetHandler.HandlePuppetUpload(
                requestData.figureId, requestData.puppetName,
                requestData.puppetJsonFile, requestData.PuppetImage,
                requestData.PuppetThumbnail);
        }
    }
    catch (Exception ex)
    {
        Debug.LogError($"Error in uploading puppet: {ex.Message}");
    }
}
```

### 6.4.8 Scene Moderation and Quality of Life Additions

Several quality-of-life enhancements have been incorporated to enhance the overall versatility of the eShadow platform. additionally a robust moderation system has been implemented to empower moderators to effectively manage and mitigate undesired behaviors, ensuring a more controlled and productive environment.

**Key Features:**

- Implements toggling of key UI elements, such as icons for QR codes, recording, and the main menu, to declutter the interface during performances, as well as allowing for performance recording through 3rd party tools.
- Introduces screen horizontal inversion command to support backprojection setups.
- Provides a moderation tool to delete unwanted objects in real-time using intuitive user input mechanisms.

- Enables rapid scene resets.

#### 6.4.8.1 Toggle UI Visibility

Key UI elements such as the QR code, recording, and main menu icons can be toggled dynamically. Activated using the **T Key** on the keyboard.

```
 if (Input.GetKeyDown(KeyCode.T))
        {
            ToggleVisibilityState();
        }
...
void ToggleVisibilityState()
{
    if (QRCodeIcon != null)
        QRCodeIcon.SetActive(!QRCodeIcon.activeSelf);

    if (RecordIcon != null)
        RecordIcon.SetActive(!RecordIcon.activeSelf);

    if (MainMenuIcon != null)
        MainMenuIcon.SetActive(!MainMenuIcon.activeSelf);
}
```

#### 6.4.8.2 Screen Flipping for Backprojection Support

The `FlipCanvas` method allows for seamless switching between regular and mirrored canvas views, catering to backprojection setups often used in shadow theater. Activated using the **F Key** on the keyboard.

```
  if (Input.GetKeyDown(KeyCode.F))
        {
            isFlipped = !isFlipped;
            // Debug.Log("Flip toggled: " + isFlipped);

            // Flip the Canvas scale
            FlipCanvas(isFlipped);
        }
...
void FlipCanvas(bool flip)
{
    if (uiCanvas != null)
    {
        Vector3 canvasScale = uiCanvas.transform.localScale;
        canvasScale.x = flip ? -Mathf.Abs(canvasScale.x) : Mathf.Abs(
            canvasScale.x);
        uiCanvas.transform.localScale = canvasScale;
```

```
17
18
19      }
20  }
```

#### 6.4.8.3 Real-Time Object Moderation

Moderation tools are provided to dynamically delete objects in the scene based on user input. The `DestroyClickedObject` method detects and removes objects under the cursor. Activated using the **ctrl + left click** with the cursor on the object to be destroyed.

```
1  if (Input.GetKey(KeyCode.LeftControl) || Input.GetKey(KeyCode.
       RightControl))
2          {
3              if (Input.GetMouseButtonDown(0) || Input.GetMouseButtonDown
                   (1))
4              {
5                  DestroyClickedObject(Input.mousePosition);
6              }
7          }
8  ...
9  GameObject DestroyClickedObject(Vector3 input)
10 {
11     Vector2 rayPos = new Vector2(
12         Camera.main.ScreenToWorldPoint(input).x,
13         Camera.main.ScreenToWorldPoint(input).y
14     );
15
16     RaycastHit2D[] hits = Physics2D.RaycastAll(rayPos, Vector2.zero, 0f);
17     foreach (var hit in hits)
18     {
19         if (hit && hit.transform && hit.transform.gameObject.CompareTag(
               Constants.SceneryTag))
20         {
21             Destroy(hit.transform.gameObject);
22             return hit.transform.gameObject;
23         }
24     }
25     return null;
26 }
```

#### 6.4.8.4 Scene Reset and Reload

A single keypress can reload the current scene, clearing it from all instatiated objects in the process. Activated using the **R Key** on the keyboard.

```
if (Input.GetKeyDown(KeyCode.R))
{
    SceneManager.LoadScene(SceneManager.GetActiveScene().name);
}
```

## 6.5 Training the LoRA Model

Transferring the art style of an image into the traditional Greek shadow theater aesthetic presented significant challenges, requiring extensive research and experimentation to achieve within the constraints of my home PC's limited hardware capabilities. Due to the lack of available pre-trained models online, it was necessary to develop one locally. The digital figures included in eShadow, created by Mr. Nikos Mplazakis, proved invaluable as a training dataset. Their digitized clarity, uniform dimensions, and consistent artistic style—resulting from being designed by the same individual—greatly enhanced the coherence of the desired aesthetic and facilitated the model training process.

Much of the experimentation was conducted using the `AUTOMATIC1111 Stable Diffusion WebUI` [2], a robust, user-friendly, and feature-rich graphical user interface (GUI) designed for interacting with Stable Diffusion models without requiring direct coding. This tool allowed for seamless experimentation with various model combinations and parameter adjustments, providing critical insights and streamlining the process of refining the methodology.

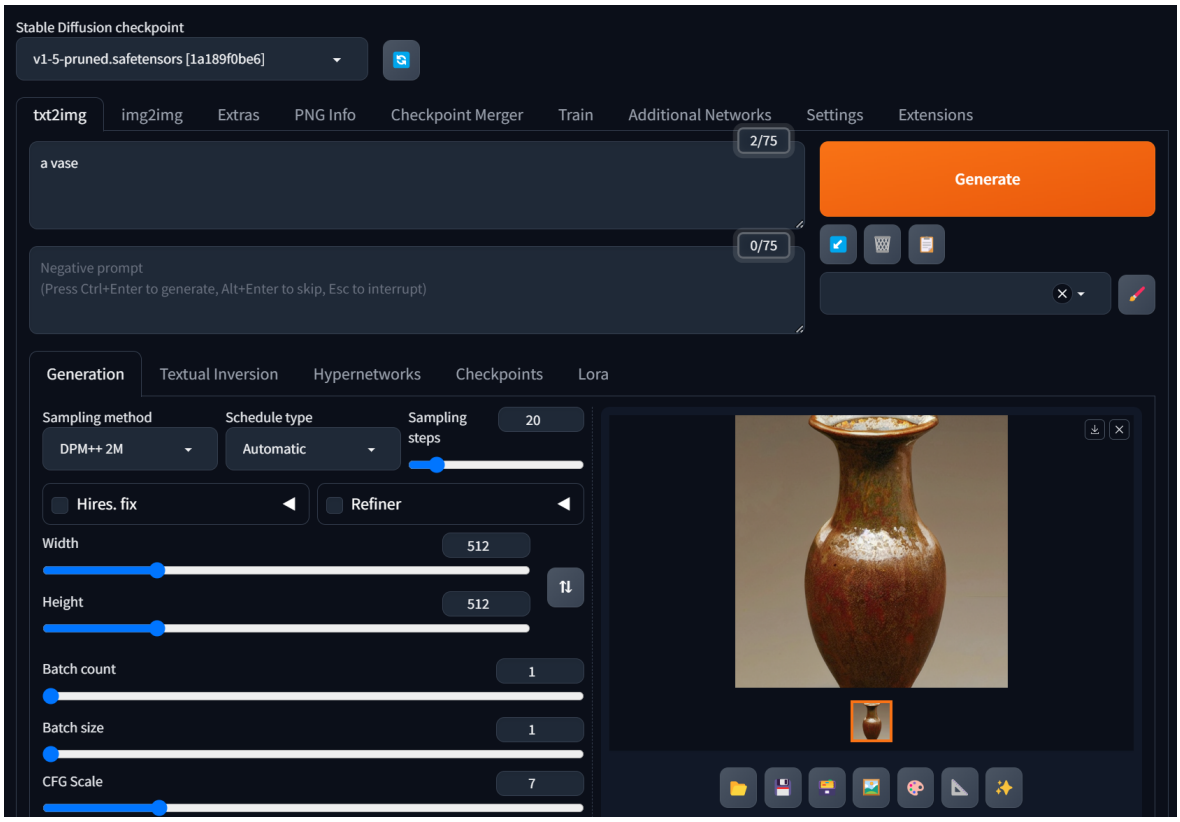



Figure 6.3: AUTOMATIC1111 Stable Diffusion WebUI

For the training of the LoRA model, the `kohya_ss` toolkit [30] was employed to streamline the process and alleviate some of the complexities involved in preparing the training pipeline. The toolkit's intuitive user interface and efficient workflow for creating LoRA models, combined with robust community support, greatly facilitated the training process. Particular gratitude is extended to Aitrepreneur's tutorial video [1], which provided valuable guidance on optimizing the training to accommodate the limited capabilities of my hardware.

The base model used for training the LoRA was `Stable Diffusion v1-5` [11], which was selected after extensive experimentation. Alternative base models, such as `majicMIX realistic` [33], were also tested. However, the resulting aesthetic from these experiments leaned more toward an anime style, which did not align with the desired outcome. Ultimately, `Stable Diffusion v1-5` was chosen due to the wide variety of images included in its training dataset, which provided greater flexibility and alignment with the intended artistic style.

Additionally, `ControlNet` was integrated into the process to preserve key characteristics of the input image. Initially, the `Canny` model [66] was utilized, but it retained too much detail, making the output overly similar to the input image. As a result, the `SoftEdge` model [67] was adopted instead. The `SoftEdge` model's more abstract outlines allowed for greater creative interpretation, enabling the model to better blend the input with the desired aesthetic.

The hardware used for training consisted of an NVIDIA RTX 4060 GPU with 8GB of VRAM, 32GB of system RAM, and an AMD Ryzen 5 5600X six-core processor. The training

process took approximately 25 minutes and was conducted using the following configuration file.


```
{
  "LoRA_type": "Standard",
  "LyCORIS_preset": "full",
  "adaptive_noise_scale": 0,
  "additional_parameters": "",
  "async_upload": false,
  "block_alphas": "",
  "block_dims": "",
  "block_lr_zero_threshold": "",
  "bucket_no_upscale": true,
  "bucket_reso_steps": 64,
  "bypass_mode": false,
  "cache_latents": true,
  "cache_latents_to_disk": false,
  "caption_dropout_every_n_epochs": 0,
  "caption_dropout_rate": 0,
  "caption_extension": ".txt",
  "clip_skip": 2,
  "color_aug": false,
  "constrain": 0,
  "conv_alpha": 1,
  "conv_block_alphas": "",
  "conv_block_dims": "",
  "conv_dim": 1,
  "dataset_config": "",
  "debiased_estimation_loss": false,
  "decompose_both": false,
  "dim_from_weights": false,
  "dora_wd": false,
  "down_lr_weight": "",
  "dynamo_backend": "no",
  "dynamo_mode": "default",
  "dynamo_use_dynamic": false,
  "dynamo_use_fullgraph": false,
  "enable_bucket": false,
  "epoch": 1,
  "extra_accelerate_launch_args": "",
  "factor": -1,
  "flip_aug": false,
  "fp8_base": false,
  "full_bf16": false,
  "full_fp16": false,
  "gpu_ids": "",
  "gradient_accumulation_steps": 1,
```

```
  "gradient_checkpointing": true,
  "huber_c": 0.1,
  "huber_schedule": "snr",
  "huggingface_path_in_repo": "",
  "huggingface_repo_id": "",
  "huggingface_repo_type": "",
  "huggingface_repo_visibility": "",
  "huggingface_token": "",
  "ip_noise_gamma": 0,
  "ip_noise_gamma_random_strength": false,
  "keep_tokens": 0,
  "learning_rate": 0.0001,
  "log_tracker_config": "",
  "log_tracker_name": "",
  "log_with": "",
  "logging_dir": "C:/Users/chris/Desktop/Karagiozis-Artstyle-Change-
     Diffusion-Model/LoRA_dataset/Karagiozis LORA/logs",
  "loss_type": "l2",
  "lr_scheduler": "constant",
  "lr_scheduler_args": "",
  "lr_scheduler_num_cycles": 1,
  "lr_scheduler_power": 1,
  "lr_warmup": 0,
  "main_process_port": 0,
  "masked_loss": false,
  "max_bucket_reso": 2048,
  "max_data_loader_n_workers": 1,
  "max_grad_norm": 1,
  "max_resolution": "512,512",
  "max_timestep": 1000,
  "max_token_length": 75,
  "max_train_epochs": 0,
  "max_train_steps": 1600,
  "mem_eff_attn": true,
  "metadata_author": "",
  "metadata_description": "",
  "metadata_license": "",
  "metadata_tags": "",
  "metadata_title": "",
  "mid_lr_weight": "",
  "min_bucket_reso": 256,
  "min_snr_gamma": 0,
  "min_timestep": 0,
  "mixed_precision": "fp16",
  "model_list": "runwayml/stable-diffusion-v1-5",
  "module_dropout": 0,
  "multi_gpu": false,
```

```
  "multires_noise_discount": 0.3,
  "multires_noise_iterations": 0,
  "network_alpha": 128,
  "network_dim": 128,
  "network_dropout": 0,
  "network_weights": "",
  "noise_offset": 0,
  "noise_offset_random_strength": false,
  "noise_offset_type": "Original",
  "num_cpu_threads_per_process": 2,
  "num_machines": 1,
  "num_processes": 1,
  "optimizer": "AdamW8bit",
  "optimizer_args": "",
  "output_dir": "C:/Users/chris/Desktop/Karagiozis-Artstyle-Change-
     Diffusion-Model/LoRA_dataset/Karagiozis LORA/model",
  "output_name": "karagiozis",
  "persistent_data_loader_workers": false,
  "pretrained_model_name_or_path": "runwayml/stable-diffusion-v1-5",
  "prior_loss_weight": 1,
  "random_crop": false,
  "rank_dropout": 0,
  "rank_dropout_scale": false,
  "reg_data_dir": "",
  "rescaled": false,
  "resume": "",
  "resume_from_huggingface": "",
  "sample_every_n_epochs": 0,
  "sample_every_n_steps": 0,
  "sample_prompts": "",
  "sample_sampler": "euler_a",
  "save_every_n_epochs": 1,
  "save_every_n_steps": 0,
  "save_last_n_steps": 0,
  "save_last_n_steps_state": 0,
  "save_model_as": "safetensors",
  "save_precision": "fp16",
  "save_state": false,
  "save_state_on_train_end": false,
  "save_state_to_huggingface": false,
  "scale_v_pred_loss_like_noise_pred": false,
  "scale_weight_norms": 0,
  "sdxl": false,
  "sdxl_cache_text_encoder_outputs": false,
  "sdxl_no_half_vae": false,
  "seed": 1234,
  "shuffle_caption": false,
```

```
  "stop_text_encoder_training_pct": 0,
  "text_encoder_lr": 5e-05,
  "train_batch_size": 1,
  "train_data_dir": "C:/Users/chris/Desktop/Karagiozis-Artstyle-Change-
     Diffusion-Model/LoRA_dataset/Karagiozis LORA/image",
  "train_norm": false,
  "train_on_input": true,
  "training_comment": "",
  "unet_lr": 0.0001,
  "unit": 1,
  "up_lr_weight": "",
  "use_cp": false,
  "use_scalar": false,
  "use_tucker": false,
  "v2": false,
  "v_parameterization": false,
  "v_pred_like_loss": 0,
  "vae": "",
  "vae_batch_size": 0,
  "wandb_api_key": "",
  "wandb_run_name": "",
  "weighted_captions": false,
  "xformers": "xformers"
}
```


The dataset used for the training included the 13 images displayed below.

Figure 6.4: Training Dataset

#### 6.5.0.1 Training Results

Figure 6.5: Input Test Image

Figure 6.6: Results Without ControlNet

Figure 6.7: Stable Diffusion 1.5 as Base Model

Figure 6.8: majicMIX as Base Model

Figure 6.9: Results With ControlNet's Softedge

As can be observed in the results the combination of Stable Diffision 1.5 [11] ControlNet's Softedge model [67] along with the localy trained LoRA [25] model yield results closest to the desired aesthetic.

## 6.6 System Deployment

To ensure the system's usability, it requires a robust method for compilation and deployment. Fortunately, both Unity and Flutter provide highly versatile compilation tools, making the process efficient and flexible. The initial target platforms for deployment were Android OS for the mobile application and Microsoft Windows for the Unity application. These platforms were selected due to their widespread popularity and ease of deployment.

The decision to target Android OS and Microsoft Windows stemmed from their alignment with the preferences of the average user, who rarely utilizes alternatives such as Linux for PC operating systems. Additionally, while Android's main competitor, iOS, is a significant player in the mobile market, its restrictive policies regarding sideloading and the substantial challenges associated with publishing apps on the Apple App Store presented significant obstacles. By contrast, publishing on the Google Play Store is comparatively straightforward, making Android the more practical choice for this project.

One of the most significant challenges was packaging the main server into a format that could bundle all dependencies, including the code and the background removal model, into a standalone executable. After thorough research, `PyInstaller` [19] was identified as the most suitable tool for the task.

The following command was used to achieve this:

Listing 6.1: Command to bundle the main server using PyInstaller.

```
pyinstaller --onefile --console --add-data "u2netp.onnx;." --distpath ../
    build_output/dist --workpath ../build_output/build server.py
```

This command successfully bundled all necessary components into a user-friendly executable file (`.exe`), ensuring ease of deployment and use.

Due to the significant complexity involved in deploying the style transformation server as a standalone portable application, it was determined that running it directly from the `PyCharm` [28] integrated development environment (IDE) would be more practical for the purposes of this thesis. Several factors influenced this decision, including the substantial hardware requirements of the server, which exceed the capabilities of most average users' PCs, even with extensive optimizations applied. Additionally, the server's large size (6.59 GB) further complicated its portability.

For future use, it is recommended that the server be hosted on a dedicated, powerful machine with a static IP address, ensuring accessibility and performance without imposing heavy hardware demands on end users.

## 6.7 Design Choices

### 6.7.1 Why Aim for Smartphones

The decision to utilize a mobile application instead of other platforms, such as desktop applications or web-based solutions, was driven by several factors. Mobile devices offer portability, ease of use, and accessibility, making them ideal for the collaborative and interactive nature of the eShadow system. The widespread availability of smartphones and their integrated features, such as cameras and touch interfaces, enables seamless functionality like image capture, puppet manipulation, and real-time interaction without requiring additional hardware.

### 6.7.2 Why Develop Using Flutter

The choice of Flutter as the development framework was based on its unique advantages over competitors like React Native or native development. Flutter's ability to build natively compiled applications for both Android and iOS from a single codebase ensures broad platform compatibility with reduced development time and cost. Its widget-based architecture allows for highly customizable and visually consistent user interfaces, ensuring an intuitive and responsive experience for users. Unlike React Native, Flutter eliminates the need for a JavaScript bridge, resulting in better performance, particularly for animation-heavy applications like eShadow. Additionally, Flutter's extensive library ecosystem and strong commu-

nity support simplified the implementation of complex functionalities such as camera integration, real-time networking, and image processing.

### 6.7.3 Translation and Localization

Translation and localization are vital components of modern software development, enabling applications to cater to diverse linguistic and cultural audiences. This project employed the `easy_localization` **easy_localization** package to implement these features effectively.

#### 6.7.3.1 Overview of Localization

Localization extends beyond language translation to include cultural adaptation, formatting standards, and regional usability requirements. For this project, ensuring accessibility for both Greek-speaking and English-speaking users as well as implementing an easy way to expand the supported languages was a primary goal.

#### 6.7.3.2 Implementation with `easy_localization`

The `easy_localization` package was selected for its simplicity and flexibility in managing multiple languages. Key features utilized include:

- **Dynamic Language Switching:** Allowing users to switch languages within the application.
- **Structured Localization Files:** Using JSON-based key-value pairs for each supported language.
- **Fallback Mechanisms:** Ensuring default messages are displayed if translations are unavailable.

The following example demonstrates the JSON structure used for the English and Greek translations:

Listing 6.2: English Translation File

```
{
  "appBarTitles": {
    "puppets": "Puppets",
    "gallery": "Gallery",
    "backgrounds": "Backgrounds"
  },
  "status": {
    "connected": "Connected to server",
    "not_connected": "Not connected to server"
  }
}
```

Listing 6.3: Greek Translation File

```
{
  "appBarTitles": {
    "puppets": "Φ       ",
    "gallery": "Σ      ",
    "backgrounds": "Σ      "
  },
  "status": {
    "connected": "Σ                                    ",
    "not_connected": "Δ                                              "
  }
}
```

One of the main challenges was ensuring that all text elements in the application were localized without hardcoding strings. This was addressed by replacing static strings with keys mapped to the corresponding translations in the JSON files. Additionally, ensuring proper alignment and rendering of Greek text required adjustments to font settings and character encoding.

### 6.7.4 Why the Style Transform Server Is Designed To Be Modular

The modular design of the style transformation server addresses critical challenges in deployment and scalability. Deploying the server locally requires substantial computational resources that are often unavailable to the average user. By adopting a modular approach, the server is decoupled from the main application, enabling it to be run independently on capable systems or centralized servers. This flexibility ensures that resource constraints do not limit access, allowing users to leverage advanced capabilities without overburdening their local hardware.

### 6.7.5 Why Package the Main Server Separately from eShadow

Separating the main server from the eShadow platform maintains the integrity and usability of the original application. This separation ensures that eShadow can operate as a standalone tool for users who prefer not to integrate additional features. Furthermore, Unity's process management can occasionally fail to terminate background processes, causing the server to continue running even after eShadow has been closed. Packaging the server separately mitigates this issue, providing a clean and isolated environment for users.

### 6.7.6 Why Use ControlNet and Why Use the Softedge Model

ControlNet was essential for retaining the key features of the input image during style transformation. The Softedge model was selected after careful experimentation, as it effectively

balances preserving critical structural elements with providing the diffusion algorithm the freedom to generate creative outputs. This balance ensures that the resulting image aligns with the traditional Greek shadow theater aesthetic without excessive resemblance to the input image.

### 6.7.7 Where to Place the Joint in the Puppet Control Part

The control part is moved using a joint attached to the mouse cursor. For touchscreens, defining this spot is challenging due to input variability. To address this, the joint is placed at a predefined location, 10% above the center of mass of the part's overall size. This configuration, determined through experimentation, has proven effective in most cases by maintaining a natural balance and ease of use.

### 6.7.8 Why Only One Figure and One Background for Each Device

Allowing each device to control only one figure and one background significantly simplifies user interaction. This decision reduces the potential for control complexity, especially for younger users, such as children. Additionally, during testing, children often overused features, filling the canvas with excessive elements and hindering collaboration. Limiting each device to one figure and one background ensures a balanced and manageable interactive experience for all participants.

### 6.7.9 Why Create a Scene Moderation System

During testing, children were observed to misuse the freedom provided by the controller, for example, resizing their figures to dominate the screen. To address this, two moderation measures were implemented. By pressing `Ctrl` and left-clicking an object, moderators can remove it. Additionally, pressing the `R` key clears the entire screen. These features maintain a structured and collaborative environment during performances.

## 6.8 Software Packaging and Distribution

The software is packaged and distributed in 3 parts. The Unity eShadow executable, the main server executable and the .apk mobile application. The style transform server was not distributed to the participants of the various activities due to not having found a clean straightforward solution to distribute it as of the time of the respective activities. Due to the modular nature of the style transform server, the system can function in a meaningful way even in its absence.

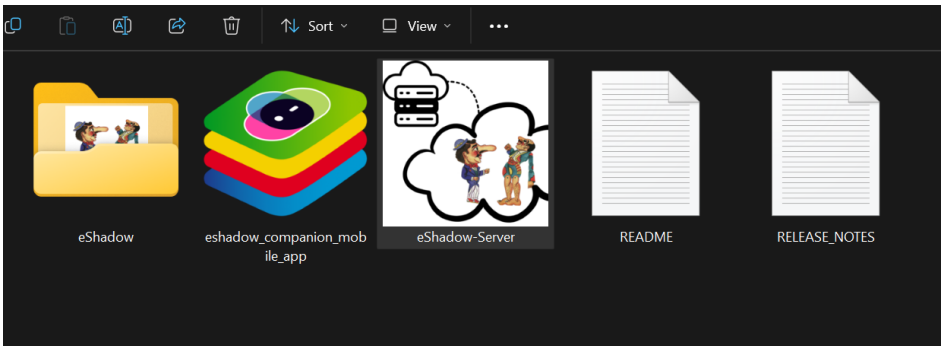



Figure 6.10: Distibuted Files

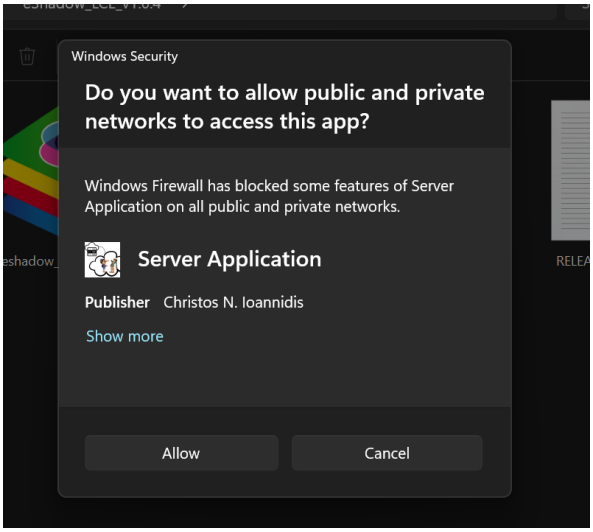



Figure 6.11: Networking Permissions are Required for the Application To Function

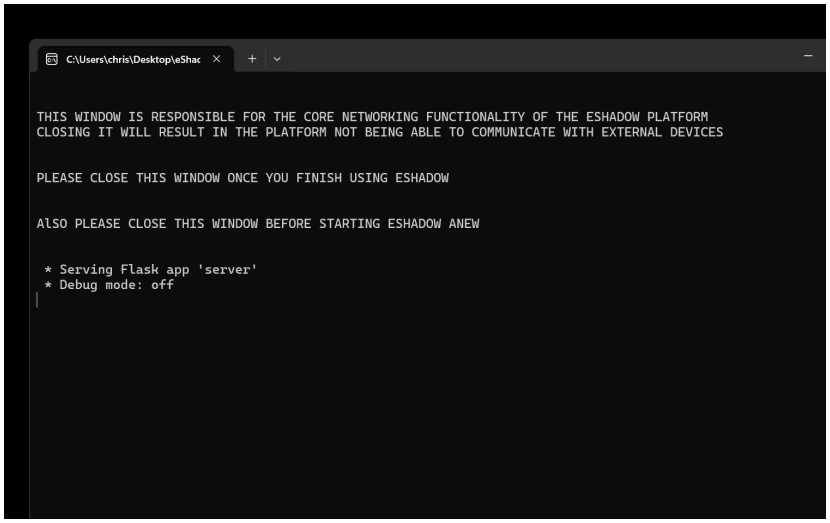



Figure 6.12: Instance of the Main Server Running

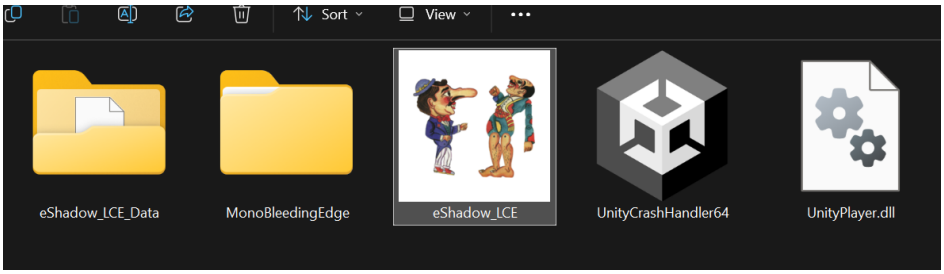



Figure 6.13: eShadow LCE File System

# 7. Evaluation

This chapter presents the evaluation of the eShadow LCE system, assessing its effectiveness in addressing the limitations identified in earlier chapters. The evaluation focuses on both technical performance and user experience, providing insights into how the new features—such as real-time collaboration and enhanced puppet creation—impact usability, engagement, and educational potential.

The evaluation process incorporates a mix of qualitative and quantitative methods. Workshops and user testing sessions were conducted with educators, students, and artists to simulate real-world use cases. Structured feedback through questionnaires and observational data were used to gauge the system's strengths and areas for improvement.

This chapter concludes with a discussion of the findings, identifying key successes and challenges. The insights gained from this evaluation inform potential improvements and guide future potential use scenarios for the eShadow LCE system.

eShadow LCE was evaluated during the course of the three events described below.

## 7.1 Chania Film Festival Workshops

### 7.1.1 About Chania Film Festival

The Chania Film Festival (CFF) is an international cinematic event held annually in Chania, Crete, typically during the last week of October. It has established itself as a significant cultural institution, featuring a diverse array of film screenings, exhibitions, masterclasses, and educational programs. The festival serves as a vibrant platform for filmmakers, artists, and audiences to engage with the art of cinema.

A cornerstone of the CFF is its comprehensive workshop offerings, designed to cater to various age groups and interests. For the educational community, the festival organizes workshops and screenings tailored for elementary, middle, and high school students. These sessions aim to cultivate an appreciation for film and enhance audiovisual literacy among young learners.

### 7.1.2 Basic Idea of the Workshop

At the 2024 Chania Film Festival, a workshop was organized as a showcase for this work, presenting its potential as an educational tool to students and educators. The workshop aimed to demonstrate how the platform could be used to enhance the learning process through interactive and collaborative storytelling.

A total of four workshop sessions were conducted, three of which were attended by elementary school students, while one was tailored for middle school students. The core concept of the workshop was to engage students in enacting an ancient Greek myth. This activity involved a debate component, encouraging critical thinking and active participation. A large piece of cloth was set, having a projector behind applying back-projection. Some of the were tasked to use the application to manipulate digital figures, while others to perform from behind the cloth appearing as shadows, interacting dynamically with the figures. This structure aimed the platform's versatility and its potential to combine technology with traditional storytelling methods, fostering both creativity and collaboration in an educational context.

### 7.1.3 The Ancient Myth Being Enacted

The myth centers on a contest between two Olympian gods, Poseidon and Athena, to determine who would become the patron deity of a prominent Greek city, later named Athens. According to the myth, the gods sought to win the favor of the city's citizens by presenting gifts that would benefit their future prosperity.

Poseidon, god of the sea, struck the ground with his trident, creating a clean water spring. Athena, goddess of wisdom and strategy, offered an olive tree.

Gods and citizens of the city were to debate over the usefulness of the two gifts and choose one.

### 7.1.4 Course of the Workshop

The workshop began with a narrated presentation that introduced the students to the myth of Poseidon and Athena, setting the stage for the activities to follow. After the presentation, the students were randomly provided with six mobile devices preloaded with the eShadow Companion app. Basic instructions were given on how the debate would unfold.

Figure 7.1: Workshop Introduction

Figure 7.2: God Poseidon and Goddess Athena

The figures of the Olympian gods were pre-created within the app, a contribution by fellow Electrical and Computer Engineering (ECE) student Mr. Christos Ioannidis, except for the figure of Poseidon, which was drawn by myself. Students with the devices were tasked with selecting their favorite god, loading the chosen figure onto the stage, and engaging in the debate, embodying the roles of the gods.


Figure 7.3: Students Controlling The Stage Figures Through the App

Simultaneously, the remaining students participated in batches of six, taking turns behind the cloth to act as citizens of the city. Their role was to debate and interact with the god figures, contributing to the myth's narrative and deepening their engagement with the story. Once each group had completed their debate behind the cloth, they were given the devices to select and act as their favorite gods, ensuring that all students had an opportunity to interact with the application and actively participate.


Figure 7.4: Students Interacting with the Stage Figures.

Figure 7.5: Students Interacting with the Stage Figures as Shadows.

This structured approach was intentionally designed to minimize idle time and ensure that every student had a meaningful and thorough interaction with the application. After the debate sessions concluded, the students were provided the opportunity to turn themselves into digital figures using the app, allowing them to interact with their classmates on stage.

Figure 7.6: Students Turning Themselves Into Figures and Playing

The workshop was warmly received by both students and educators. Many educators expressed interest in using the app as a tool to enhance classroom learning. Before concluding the workshop, participants who interacted with the app were invited to complete questionnaires evaluating their experience with the app.

## 7.2 Chania International Digital Training Workshop

### 7.2.1 About the International Digital Training

The International Digital Training, held in Chania, Crete, from October 22nd to 25th, 2024, was organized by the International Yehudi Menuhin Foundation (IYMF) in collaboration with the Technical University of Crete (TUC) and the MUS-E® Network. This event highlighted the intersections of technology and art, offering hands-on experiences to international artists, local teachers, and participants. The training showcased creative approaches to combining traditional and digital media through workshops on gamification, interactive media, and virtual reality.

### 7.2.2 Presentation of eShadow LCE

Among these sessions, the opportunity was given to present eShadow LCE and its capabilities to the participants and collect feedback.

Figure 7.7: Presenting Before the Artist Group

The workshop was met with a heartwarming reception, with participants actively engaging and posing insightful questions, including inquiries into the technical aspects of the platform. Some attendees had prior experience with the original platform and expressed enthusiastic support for the eShadow LCE system, with several providing their contact information to stay updated on the official release. To gather structured feedback, questionnaires were distributed among participants.

## 7.3 TUC Science Day 2024

### 7.3.1 About TUC Science Day

The Science and Technology Day at the Technical University of Crete is an annual event primarily designed for primary school children. It aims to introduce young students to the fields of science and engineering through hands-on, interactive activities. Under the guidance of experienced collaborators and volunteer students, participants engage in various experiments that yield impressive or seemingly ”magical” results. Additionally, they have the opportunity to observe technological demonstrations utilizing research models and tools from the university.

This event represents the university’s largest outreach initiative, featuring activities and booths from laboratories and research teams across all its schools. Supported by numerous volunteer students, the event ensures a smooth and enriching experience for all attendees. By integrating the four STEM fields—Science, Technology, Engineering, and Mathematics —the event fosters critical thinking, problem-solving, and innovation among young learners. Through interactive experiments and technological showcases, children are encouraged to ”discover” new knowledge, with the ultimate goal of cultivating a research-oriented mindset and attitude.

During this event, a designated stand was assigned, where an interactive experience was designed and facilitated to showcase the eShadow LCE system capabilities to the visitors.

### 7.3.2 Designing an Interactive Experience

Creating an engaging interactive experience for a continuous flow of visitors with relatively short visit durations proved to be a challenging task. The primary objective was to leverage the widespread availability of mobile phones to fully showcase the capabilities of the application.

The concept involved providing users with drawing templates, which they could fill using markers. These drawings were then scanned using the application and loaded onto a stage, represented by a large cloth. Once loaded, visitors could interact with their creations collaboratively, engaging with the designs created by others.

For those without mobile phones, or for iOS users (as the app was not supported on iOS at the time), six mobile devices preloaded with the application were made available. These devices were also connected to the style transfer server, offering additional functionality. Additionally, QR codes were posted on various walls throughout the event venue, allowing visitors to easily download the application on their own devices.

The stand was manned with seven volunteers who ensured the smooth flow of visitors, assisted with app installations, and provided guidance on using the application.

The drawing templates used during the event are presented below.

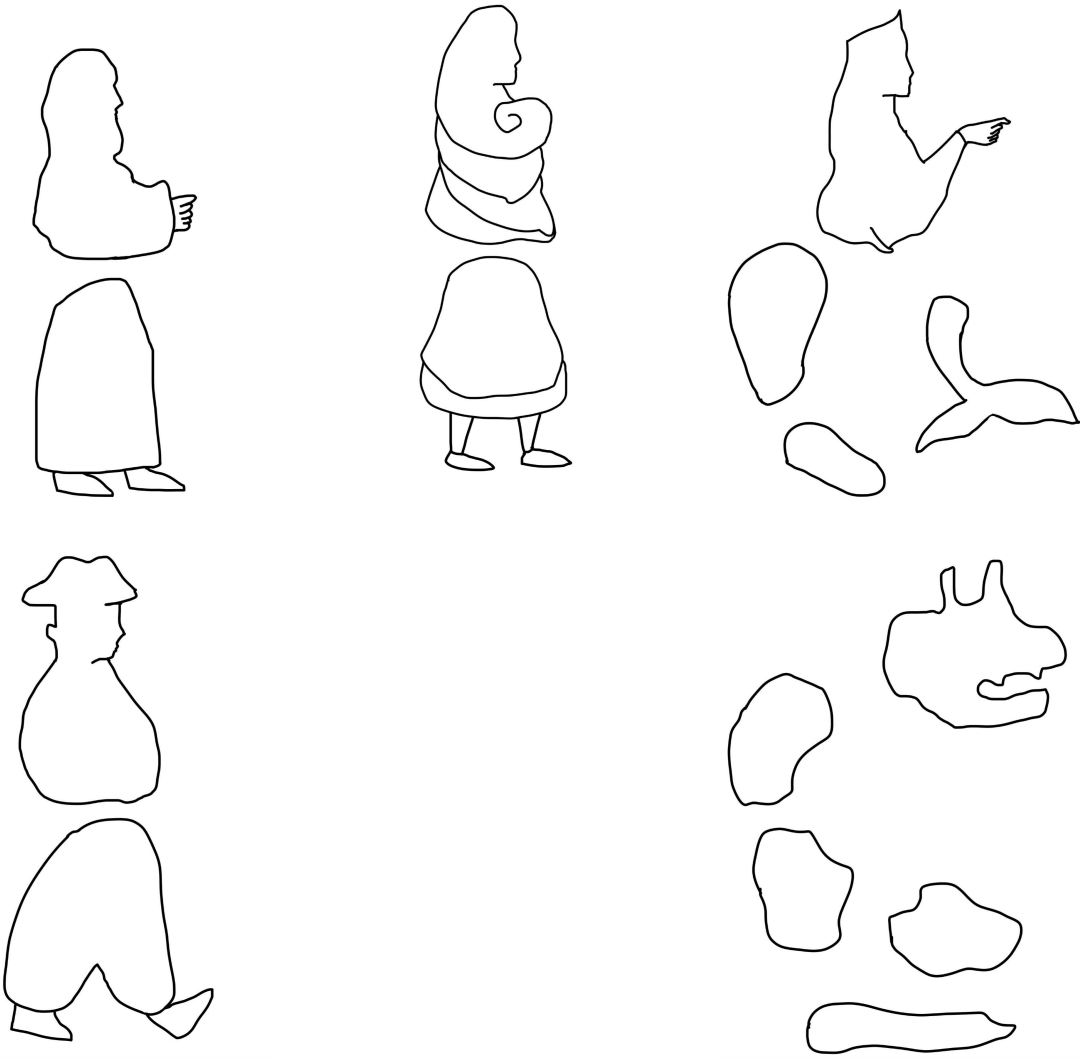
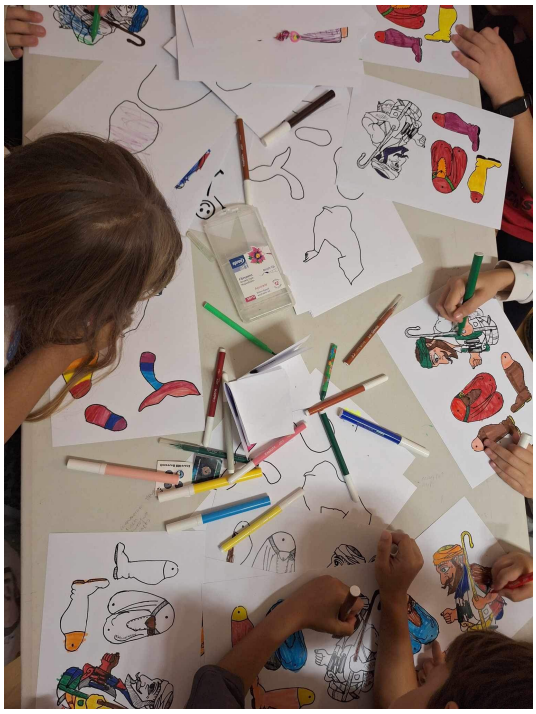
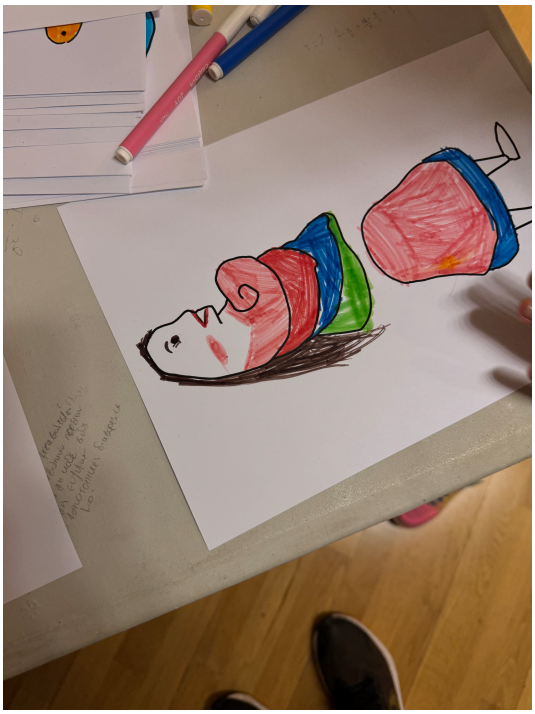


Figure 7.8: Drawing Templates Given to Visitors

(a) Preparation of the Drawing Table



(b) Drawing Table in Action



(c) Sample Creation

(d) Drawn Figures Loaded to Digital Stage

Figure 7.9: Characteristic Shots from the Event

Figure 7.10: People Playing With Their Digital Figures

To enhance the overall experience and address the time required for children to draw and scan figures, a second stage was set up near the eShadow stage. On this stage, my talented fellow ECE student, Mr. Christos Ioannidis, performed traditional Greek shadow theater. He also displayed authentic shadow figures, explaining their inner mechanisms to visitors. This dual approach not only entertained parents while their children designed and interacted with their creations but also raised awareness about traditional shadow theater, leaving a cultural impression on the children who stayed to watch the performances.

Figure 7.11: Showcase of Real Traditional Shadow Theater

The event was met with overwhelmingly positive reception, as children experienced both the past and the future of shadow theater in the same space, while adults revisited nostalgic memories. Numerous attendees, particularly educators, expressed interest in the application and provided their contact information to be notified upon its official release.

To further evaluate the experience, some volunteers were tasked with observing visitors who extensively used the app and distributing evaluation forms. They also assisted the visitors in completing these forms.

## 7.4 Quantitative Evaluation of eShadow LCE

Accurate and reliable evaluation of the application was a key objective throughout the process. To achieve this, standardized and scientifically validated questionnaires, specifically designed for evaluating similar applications, were employed.

Across all the aforementioned events, a total of **142** questionnaires were completed by participants. It is important to note that the actual number of individuals who engaged with the eShadow LCE system is likely significantly higher. However, due to constraints in time and human resources—particularly during the TUC Science Day—the focus was placed on collecting responses from participants who had sufficient time to interact meaningfully with the application. This approach ensured that the data reflected the experiences of users who had an in-depth engagement with the platform, resulting in the final dataset of 142 questionnaires. The following sections provide a thorough explanation of the questionnaires and a detailed analysis of the collected data. It is worth noting that between the Chania Film Festival workshops and the subsequent activities, a time interval of several weeks was dedicated to an iterative improvement process. During this period, issues observed during the Chania Film Festival workshops, along with feedback provided by users, were carefully documented

and compiled into a comprehensive list. Most of these issues were resolved before the next set of activities, which included the artist group workshop and the TUC Science Day event. As a result, the analysis has been divided into two phases: **Experiment 1**, encompassing the Chania Film Festival workshops, and **Experiment 2**, including the artist group workshop and the TUC Science Day. As will be demonstrated later, this iterative approach resulted in a significant improvement in the outcomes of Experiment 2 compared to those of Experiment 1.

### 7.4.1 User Experience Questionnaire (UEQ)

The User Experience Questionnaire (UEQ) [52] is a widely used tool designed to evaluate the user experience of interactive systems. It provides a comprehensive assessment by capturing subjective impressions across six key dimensions:

- **Attractiveness**: Measures the overall appeal and likeability of the product.
- **Perspicuity**: Assesses how intuitive and easy to learn the product is.
- **Efficiency**: Evaluates how effectively the product supports task performance.
- **Dependability**: Considers how predictable and secure the interaction feels.
- **Stimulation**: Captures the excitement and motivation provided by using the product.
- **Novelty**: Assesses the innovativeness and creative design of the product.

#### 7.4.1.1 Score Range

The UEQ employs a scale ranging from **-3 (extremely bad)** to **+3 (extremely good)**:

- **Values greater than 0.8:** Indicate a **positive evaluation**.
- **Values between -0.8 and 0.8:** Represent a **neutral evaluation**.
- **Values less than -0.8:** Indicate a **negative evaluation**.

#### 7.4.1.2 Context for Interpretation

In practical applications, the observed range of scores is often narrower than the theoretical range due to differences in user behavior and response tendencies. Specifically:

- Scores between +1.5 and +2.5 are considered **very good**.
- Scores above +2.5 are rare and denote an **exceptional user experience**.
- Scores below -2.0 are also rare and suggest significant dissatisfaction.

This study employed the UEQ to evaluate the user experience of the enhanced eShadow platform. Participants completed the UEQ after interacting with the system, providing valuable quantitative insights into their experience. The structured nature of the UEQ allows for both an overview of user satisfaction and detailed analysis of specific strengths and weaknesses. The standardized questionnaire ensures reliability and comparability of results across various contexts. For further details, refer to the official handbook available at `https://www.ueq-online.org/`.

### 7.4.2 UEQ Results

This section is divided into three main parts. The first part discusses the results from the workshops held at schools during the Chania Film Festival (Experiment 1). The second part examines the feedback gathered from the artist group and participants of the Science Day event (Experiment 2). The third part compares the findings from the first two groups.

It is important to note that out of the **72** questionnaires distributed, **23** were excluded due to inconsistencies in three or more dimensions. As a result, a total of **49** valid questionnaires were included in the final analysis.

#### 7.4.2.1 Experiment 1 (31 Questionnaires)

Table 7.1: UEQ Questionnaire Results for Chania Film Festival Workshops

| Dimension | Average | Variance |
|---|---|---|
| Attractiveness | 2.167 | 0.68 |
| Perspicuity | 1.605 | 1.61 |
| Efficiency | 1.532 | 1.00 |
| Dependability | 0.978 | 0.94 |
| Stimulation | 1.446 | 1.58 |
| Novelty | 1.554 | 1.14 |

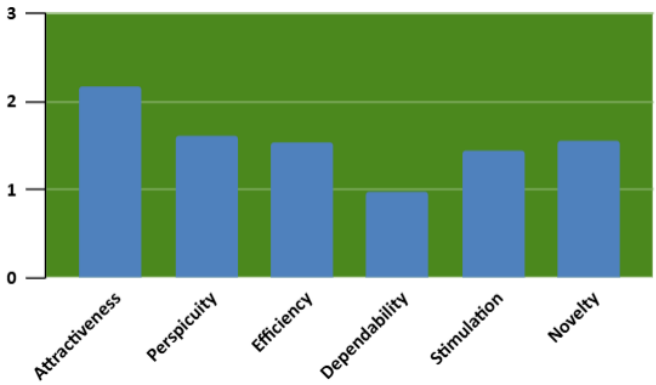



Figure 7.12: UEQ Dimension Scores for the eShadow Platform

**Interpretation:** The table summarizes the results of the UEQ questionnaires filled by the participants of the Chania Film Festival workshops. Below is an interpretation of the results:

- **Attractiveness (2.167, 0.68):** The application scored highly in attractiveness, indicating users found it visually appealing and enjoyable to use. The low variance (0.68) suggests consistent user agreement.
- **Perspicuity (1.605, 1.61):** The application was rated as moderately intuitive and easy to understand. However, the higher variance (1.61) shows that user experiences varied significantly.
- **Efficiency (1.532, 1.00):** Users rated the application as fairly efficient in terms of achieving goals with minimal effort. The variance (1.00) indicates moderate agreement among users.
- **Dependability (0.978, 0.94):** Dependability received a lower score, suggesting users found some issues with reliability or predictability. The variance (0.94) reflects moderate consistency in user feedback.
- **Stimulation (1.446, 1.58):** The stimulation score indicates the application was moderately engaging and motivating. The high variance (1.58) highlights differing user opinions.
- **Novelty (1.554, 1.14):** Novelty was rated moderately high, suggesting users found the application somewhat innovative and unique. The variance (1.14) indicates a fair level of agreement.

Overall, the results indicate that the application is perceived as attractive and relatively easy to use, though there is room for improvement in areas such as dependability and stimulation. The variances in responses suggest that user opinions were more divided, particularly regarding perspicuity and stimulation.

The modest reception in dimensions like perspicuity, efficiency, and stimulation, as well as the high variance in responses, can likely be attributed to the time constraints of the workshop. The application is designed with a moderate learning curve due to its extensive set of features, making it better suited for classroom settings where users have sufficient time to familiarize themselves with the software and experiment in a comfortable environment. Additionally, the fact that most participants were elementary school students may have contributed to the mixed results, as their cognitive abilities are still developing, which could have made it more challenging for them to fully grasp the application's functionality.

Multiple dependability issues were identified through observing users interact with the application and gathering their feedback through direct questioning, which contributed to the

low score in this dimension. These issues were meticulously documented for future corrections and improvements.

**Benchmarking:** Based on UEQ benchmarks derived from diverse datasets:

- Scores in the top 10% are classified as **Excellent**.
- Scores in the next 25% are classified as **Good**.
- Scores between 50–75% are classified as **Above Average**.
- Scores between 25–50% are classified as **Below Average**.
- Scores in the bottom 25% are classified as **Poor**.

| Scale | Mean | Comparisson to benchmark | Interpretation |
|---|---|---|---|
| **Attractiveness** | 2.17 | **Excellent** | In the range of the 10% best results |
| **Perspicuity** | 1.60 | **Above Average** | 25% of results better, 50% of results worse |
| **Efficiency** | 1.53 | **Good** | 10% of results better, 75% of results worse |
| **Dependability** | 0.98 | **Below Average** | 50% of results better, 25% of results worse |
| **Stimulation** | 1.45 | **Good** | 10% of results better, 75% of results worse |
| **Novelty** | 1.55 | **Good** | 10% of results better, 75% of results worse |

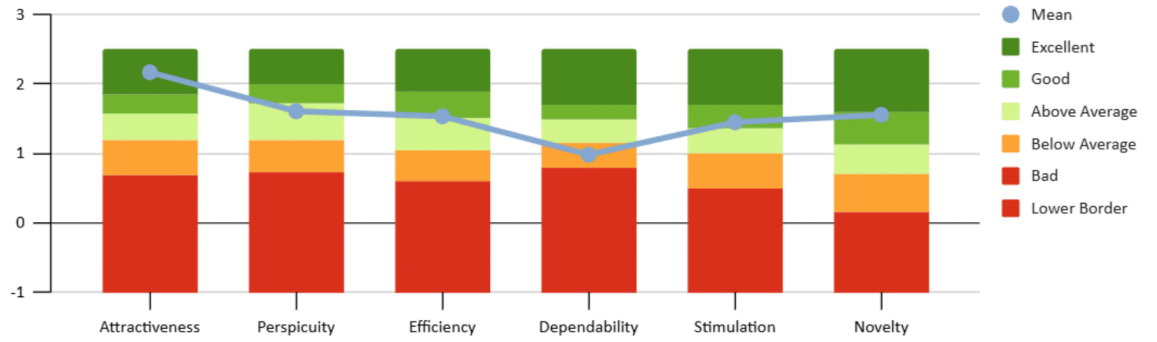



Figure 7.13: Benchmark Comparison for UEQ Scores

The benchmark results from the Chania Film Festival workshops present a mixed picture of the eShadow platform's performance. While certain dimensions showcase excellence, others reveal areas for potential improvement, highlighting the need for further refinement.

The platform scores particularly high in **Attractiveness (2.17)**, reflecting its visually appealing and engaging design. Similarly, **Perspicuity (1.60)**, **Stimulation (1.45)**, and **Novelty (1.55)** demonstrate its ability to deliver an experience that is relatively clear, engaging, and innovative. These scores validate the platform's appeal and its capacity to captivate users with its creative elements.

However, the scoring especially on **Dependability (0.98)** highlight areas requiring attention. While still positively rated, they suggest potential usability challenges and reliability concerns that could impact overall user satisfaction.

In summary, the results from this experiment underscore the platform's strengths in visual and experiential qualities, while also pointing to task-related dimensions that need enhancement. Striking a better balance between these aspects could further elevate the platform' s effectiveness and ensure a consistently positive user experience across all dimensions.

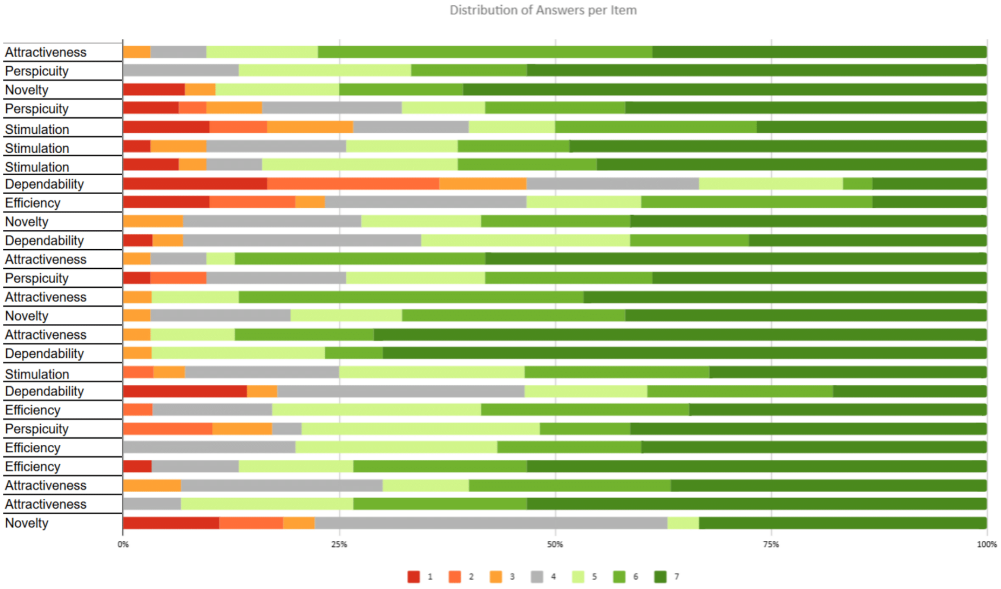



Figure 7.14: Per Item Answer Distribution

| Item | Mean | Variance | Std. Dev. | No. | Left | Right | Scale |
|---|---|---|---|---|---|---|---|
| 1 | 2.0 | 1.1 | 1.0 | 31 | annoying | enjoyable | Attractiveness |
| 2 | 2.1 | 1.3 | 1.1 | 30 | not understandable | understandable | Perspicuity |
| 3 | 2.0 | 3.0 | 1.7 | 28 | creative | dull | Novelty |
| 4 | 1.4 | 3.6 | 1.9 | 31 | easy to learn | difficult to learn | Perspicuity |
| 5 | 0.8 | 4.1 | 2.0 | 30 | valuable | inferior | Stimulation |
| 6 | 1.7 | 2.6 | 1.6 | 31 | boring | exciting | Stimulation |
| 7 | 1.7 | 2.8 | 1.7 | 31 | not interesting | interesting | Stimulation |
| 8 | -0.4 | 3.9 | 2.0 | 30 | unpredictable | predictable | Dependability |
| 9 | 0.5 | 3.6 | 1.9 | 30 | fast | slow | Efficiency |
| 10 | 1.7 | 1.9 | 1.4 | 29 | inventive | conventional | Novelty |
| 11 | 1.2 | 2.2 | 1.5 | 29 | obstructive | supportive | Dependability |
| 12 | 2.3 | 1.1 | 1.0 | 31 | good | bad | Attractiveness |
| 13 | 1.5 | 2.9 | 1.7 | 31 | complicated | easy | Perspicuity |
| 14 | 2.3 | 0.8 | 0.9 | 30 | unlikable | pleasing | Attractiveness |
| 15 | 1.9 | 1.5 | 1.2 | 31 | usual | leading edge | Novelty |
| 16 | 2.5 | 0.9 | 0.9 | 31 | unpleasant | pleasant | Attractiveness |
| 17 | 2.4 | 1.1 | 1.0 | 30 | secure | not secure | Dependability |
| 18 | 1.5 | 2.0 | 1.4 | 28 | motivating | demotivating | Stimulation |
| 19 | 0.6 | 3.6 | 1.9 | 28 | meets expectations | does not meet expectations | Dependability |
| 20 | 1.7 | 1.7 | 1.3 | 29 | inefficient | efficient | Efficiency |
| 21 | 1.4 | 2.9 | 1.7 | 29 | clear | confusing | Perspicuity |
| 22 | 1.8 | 1.4 | 1.2 | 30 | impractical | practical | Efficiency |
| 23 | 2.0 | 2.0 | 1.4 | 30 | organized | cluttered | Efficiency |
| 24 | 1.6 | 1.9 | 1.4 | 30 | attractive | unattractive | Attractiveness |
| 25 | 2.2 | 1.0 | 1.0 | 30 | friendly | unfriendly | Attractiveness |
| 26 | 0.5 | 4.3 | 2.1 | 27 | conservative | innovative | Novelty |

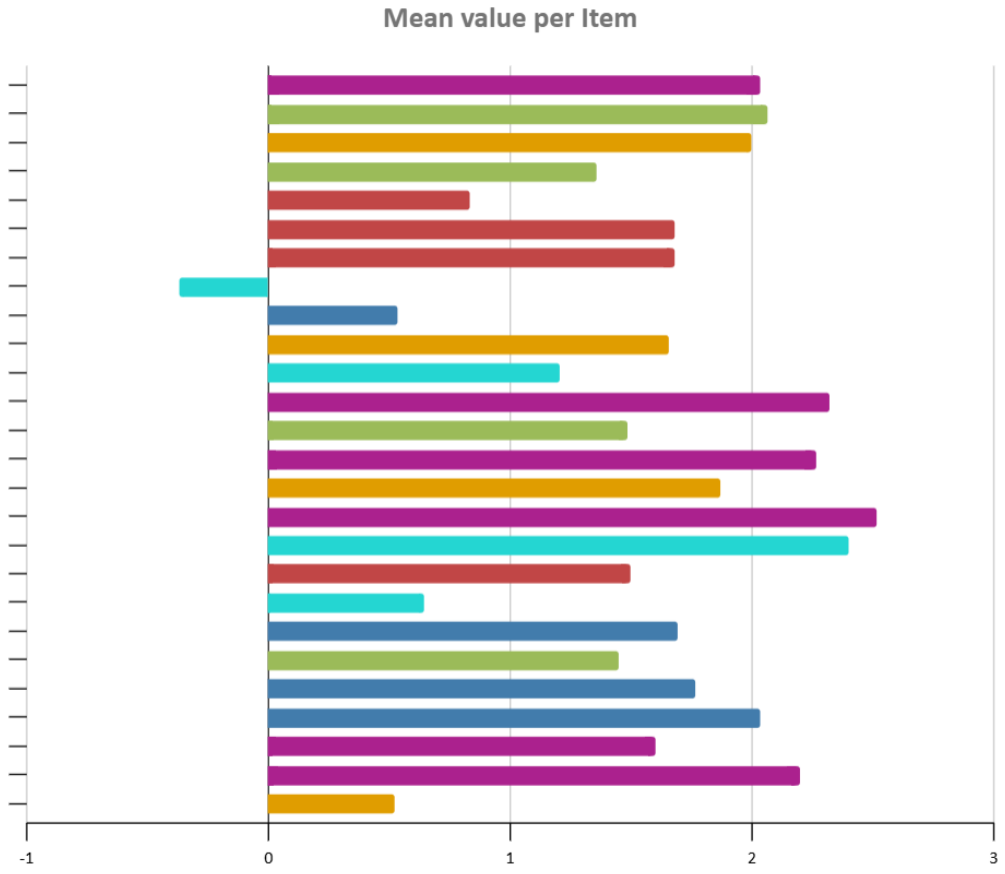



Figure 7.15: Overal Results per Item on Scale -3 , +3

#### 7.4.2.2 Experiment 2 (18 Questionnaires)

Table 7.2: UEQ Questionnaire Results for Chania international digital training workshop-Science Day Workshop

| Dimension | Average | Variance |
|---|---|---|
| Attractiveness | 2.602 | 0.45 |
| Perspicuity | 1.912 | 0.82 |
| Efficiency | 1.917 | 0.94 |
| Dependability | 1.778 | 0.70 |
| Stimulation | 2.477 | 0.71 |
| Novelty | 2.185 | 0.86 |

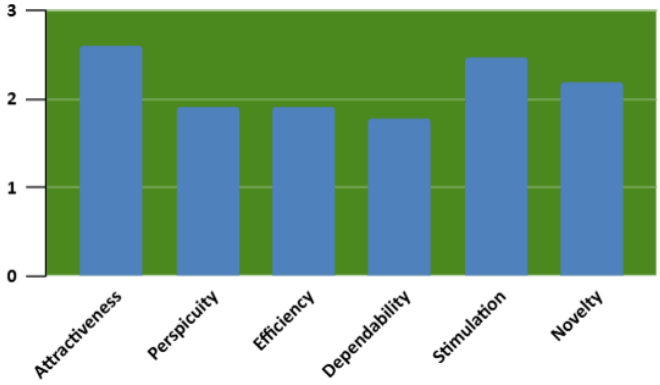



Figure 7.16: UEQ Dimension Scores for the eShadow Platform

**Interpretation:** The table presents the results of the UEQ questionnaires filled by the participants of the Chania international digital training workshop as well as the Technical University of Crete science day event. During the science day event, volunteers were instructed to hand over. Below is the detailed interpretation:

- **Attractiveness (2.602, 0.45):** The highest score among all dimensions, indicating that users found the application highly appealing and enjoyable. The low variance (0.45) reflects strong agreement among users.
- **Perspicuity (1.912, 0.82):** Users found the application relatively easy to learn and understand. The moderate variance (0.82) shows a fair level of agreement.
- **Efficiency (1.917, 0.94):** Efficiency was rated positively, indicating that users could achieve their goals with minimal effort. The variance (0.94) suggests moderate agreement among users.

- **Dependability (1.778, 0.70):** Dependability received a slightly lower score, implying users felt the application's reliability could be improved. The low variance (0.70) shows consistent feedback.
- **Stimulation (2.477, 0.71):** Stimulation scored highly, indicating the application was engaging and motivating. The low variance (0.71) highlights consistent user responses.
- **Novelty (2.185, 0.86):** The novelty score suggests that users found the application innovative and unique. The moderate variance (0.86) reflects a fair level of agreement.

Overall, the application was perceived as attractive, stimulating, and novel, with high consistency in these dimensions. The low variances in attractiveness, stimulation, and dependability highlight strong user agreement in these areas.

**Benchmarking:** Based on UEQ benchmarks derived from diverse datasets:

- Scores in the top 10% are classified as **Excellent**.
- Scores in the next 25% are classified as **Good**.
- Scores between 50–75% are classified as **Above Average**.
- Scores between 25–50% are classified as **Below Average**.
- Scores in the bottom 25% are classified as **Poor**.

| Scale | Mean | Comparisson to benchmark | Interpretation |
|---|---|---|---|
| Attractiveness | 2.60 | Excellent | In the range of the 10% best results |
| Perspicuity | 1.91 | Good | 10% of results better, 75% of results worse |
| Efficiency | 1.92 | Excellent | In the range of the 10% best results |
| Dependability | 1.78 | Excellent | In the range of the 10% best results |
| Stimulation | 2.48 | Excellent | In the range of the 10% best results |
| Novelty | 2.19 | Excellent | In the range of the 10% best results |

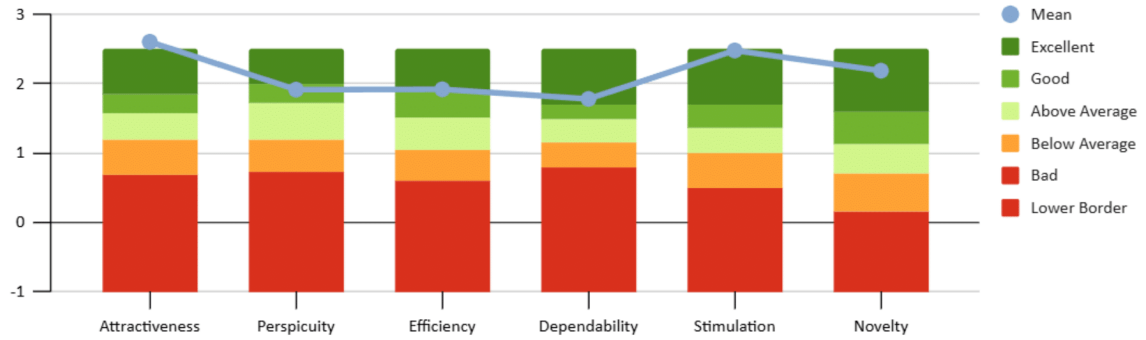



Figure 7.17: Benchmark Comparison for UEQ Scores

The benchmark results from the second experiment highlight significant improvements in the eShadow platform, with the majority of dimensions achieving excellent ratings and

only one dimension rated as good. This demonstrates the platform's growing effectiveness in delivering a positive user experience.

**Attractiveness (2.60)** emerged as the highest-rated dimension, showcasing the platform's strong visual appeal and enjoyable user experience. The low variance (0.45) indicates a high level of agreement among participants, further affirming its appeal.

**Stimulation (2.48)** and **Novelty (2.19)** were also rated as excellent, emphasizing the platform's engaging and motivating nature, as well as its innovative qualities. Both dimensions had low to moderate variance, reflecting consistent feedback from users.

**Efficiency (1.92)** and **Dependability (1.78)** scored excellently as well, indicating that users found the platform efficient in helping them achieve their goals and reasonably reliable. The slightly lower score in dependability suggests room for further refinement in ensuring consistent reliability.

**Perspicuity (1.91)**, while still rated positively as good, scored slightly lower than the other dimensions. This suggests that some users found the platform relatively easy to understand and learn, but there is still an opportunity to improve its intuitiveness to better cater to diverse user needs.

**Summary:** Overall, the results from the second experiment reflect a significant enhancement in the eShadow platform's ability to deliver a compelling and engaging user experience. The near-universal excellence across dimensions demonstrates the platform's potential as a robust and innovative tool for digital shadow theater, with minor areas for further improvement.

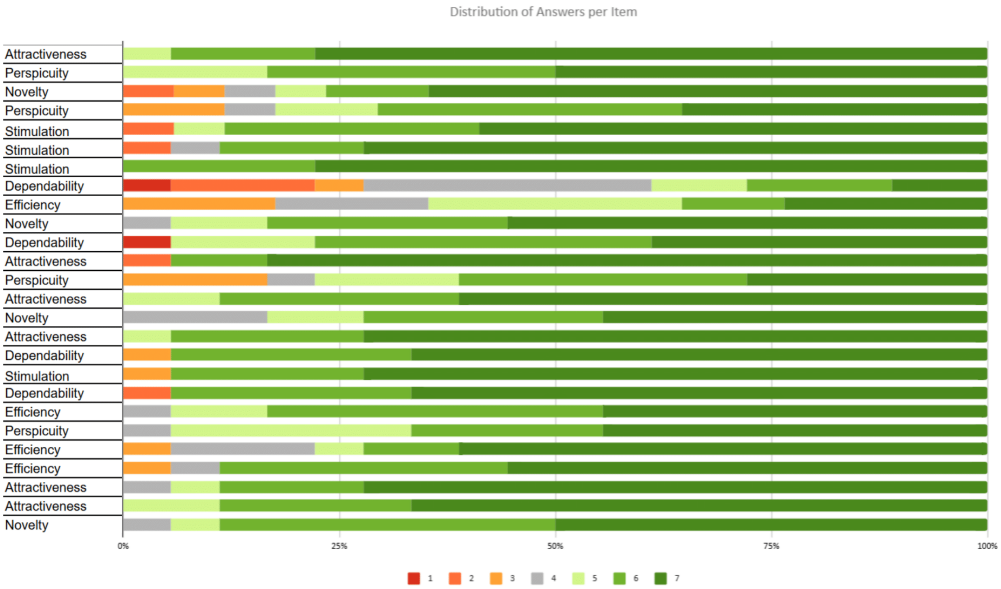



Figure 7.18: Per Item Answer Distribution

| Item | Mean | Variance | Std. Dev. | No. | Left | Right | Scale |
|---|---|---|---|---|---|---|---|
| 1 | 2.7 | 0.3 | 0.6 | 18 | annoying | enjoyable | Attractiveness |
| 2 | 2.3 | 0.6 | 0.8 | 18 | not understandable | understandable | Perspicuity |
| 3 | 2.1 | 2.6 | 1.6 | 17 | creative | dull | Novelty |
| 4 | 1.8 | 1.8 | 1.3 | 17 | easy to learn | difficult to learn | Perspicuity |
| 5 | 2.3 | 1.6 | 1.3 | 17 | valuable | inferior | Stimulation |
| 6 | 2.4 | 1.8 | 1.3 | 18 | boring | exciting | Stimulation |
| 7 | 2.8 | 0.2 | 0.4 | 18 | not interesting | interesting | Stimulation |
| 8 | 0.2 | 3.1 | 1.8 | 18 | unpredictable | predictable | Dependability |
| 9 | 1.1 | 2.1 | 1.4 | 17 | fast | slow | Efficiency |
| 10 | 2.3 | 0.8 | 0.9 | 18 | inventive | conventional | Novelty |
| 11 | 1.9 | 2.1 | 1.4 | 18 | obstructive | supportive | Dependability |
| 12 | 2.6 | 1.4 | 1.2 | 18 | good | bad | Attractiveness |
| 13 | 1.5 | 2.0 | 1.4 | 18 | complicated | easy | Perspicuity |
| 14 | 2.5 | 0.5 | 0.7 | 18 | unlikable | pleasing | Attractiveness |
| 15 | 2.0 | 1.3 | 1.1 | 18 | usual | leading edge | Novelty |
| 16 | 2.7 | 0.4 | 0.6 | 18 | unpleasant | pleasant | Attractiveness |
| 17 | 2.5 | 1.0 | 1.0 | 18 | secure | not secure | Dependability |
| 18 | 2.6 | 1.0 | 1.0 | 18 | motivating | demotivating | Stimulation |
| 19 | 2.4 | 1.4 | 1.2 | 18 | meets expectations | does not meet expectations | Dependability |
| 20 | 2.2 | 0.8 | 0.9 | 18 | inefficient | efficient | Efficiency |
| 21 | 2.1 | 1.0 | 1.0 | 18 | clear | confusing | Perspicuity |
| 22 | 2.1 | 1.9 | 1.4 | 18 | impractical | practical | Efficiency |
| 23 | 2.3 | 1.3 | 1.1 | 18 | organized | cluttered | Efficiency |
| 24 | 2.6 | 0.7 | 0.9 | 18 | attractive | unattractive | Attractiveness |
| 25 | 2.6 | 0.5 | 0.7 | 18 | friendly | unfriendly | Attractiveness |
| 26 | 2.3 | 0.7 | 0.8 | 18 | conservative | innovative | Novelty |

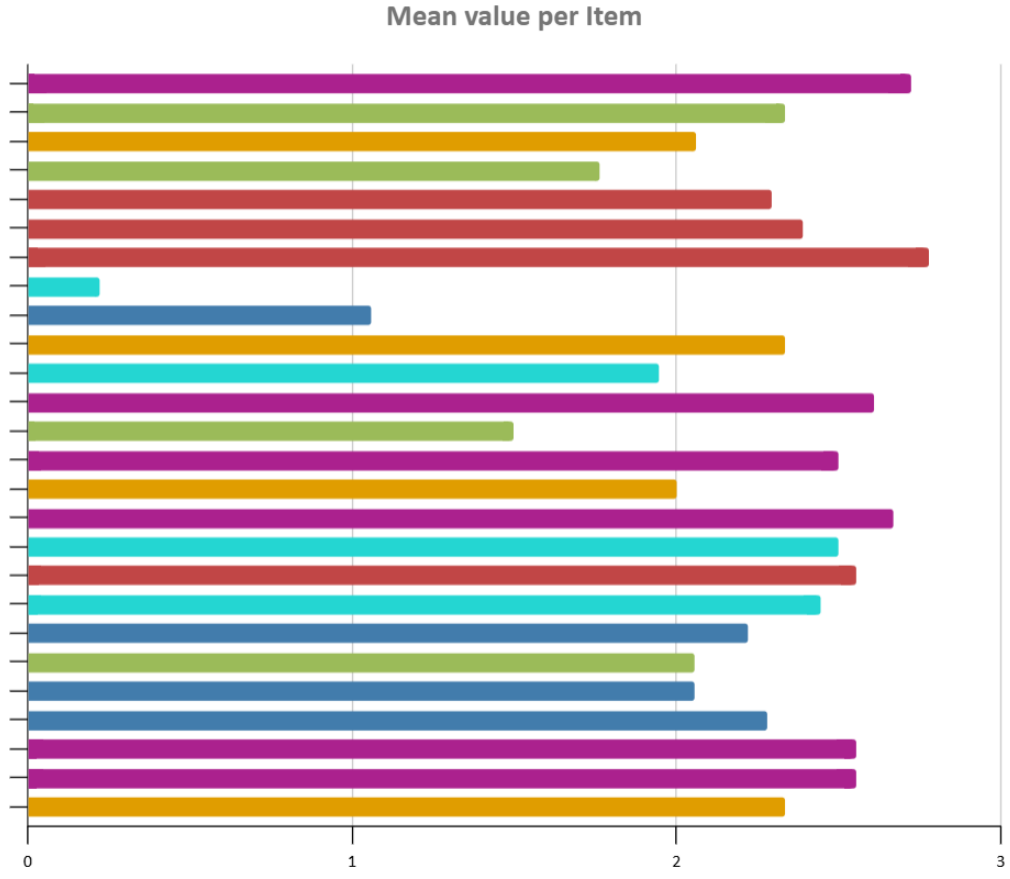



Figure 7.19: Overal Results per Item on Scale -3 , +3

#### 7.4.2.3 Results Comparison

Table 7.3: Comparison of UEQ Results Between Experiments

| Dimension | Avg. (Exp. 1) | Var. (Exp. 1) | Avg. (Exp. 2) | Var. (Exp. 2) |
|---|---|---|---|---|
| Attractiveness | 2.167 | 0.68 | 2.602 | 0.45 |
| Perspicuity | 1.605 | 1.61 | 1.912 | 0.82 |
| Efficiency | 1.532 | 1.00 | 1.917 | 0.94 |
| Dependability | 0.978 | 0.94 | 1.778 | 0.70 |
| Stimulation | 1.446 | 1.58 | 2.477 | 0.71 |
| Novelty | 1.554 | 1.14 | 2.185 | 0.86 |

**Interpretation:** The table presents the comparison of the UEQ questionnaire results between the Chania Film Festival workshops (Experiment 1) and the Science Day - Chania International Digital Training Workshop events (Experiment 2). Below is the detailed interpretation of the results:

- **Attractiveness (Experiment 1: 2.167, 0.68 | Experiment 2: 2.602, 0.45):** The attractiveness score improved significantly. The lower variance in Experiment 2 (0.45) indicates greater consistency in user agreement.
- **Stimulation (Experiment 1: 1.446, 1.58 | Experiment 2: 2.477, 0.71):** Stimulation showed substantial improvement, highlighting enhanced user engagement and motivation in the second workshop. The sharp decrease in variance also suggests that the experience was more universally positive.
- **Dependability (Experiment 1: 0.978, 0.94 | Experiment 2: 1.778, 0.70):** The dependability score increased markedly, suggesting users found the application to be more reliable and predictable. The reduced variance further indicates a more consistent perception of this dimension.
- **Efficiency (Experiment 1: 1.532, 1.00 | Experiment 2: 1.917, 0.94):** Efficiency scores improved moderately, indicating that users found it easier to accomplish tasks with the application. The slightly reduced variance suggests more uniform feedback during the second event.
- **Perspicuity (Experiment 1: 1.605, 1.61 | Experiment 2: 1.912, 0.82):** Users found the application more intuitive and easier to learn in the second workshop, as reflected by the improved score and significantly reduced variance.
- **Novelty (Experiment 1: 1.554, 1.14 | Experiment 2: 2.185, 0.86):** The novelty dimension also showed improvement, indicating that participants viewed the application

as more innovative in Experiment 2. The reduced variance highlights more consistent feedback in this area.

**Summary:** Overall, the results from the Science Day - Chania International Digital Training Workshop events show significant improvements across all dimensions compared to the earlier Chania Film Festival workshops. Notably, scales such as perspicuity and efficiency scored much higher as observations recorded in Experiment 1 helped significantly in identify and resolve issues related to the application's flow. Additionally the biggest improvement can be observed in the dependability scale, something made possible again by the iterative development process followed to identify and fix key issues.

### 7.4.3 System Usability Scale (SUS)

The System Usability Scale (SUS)[5] is a robust and reliable tool widely used to measure the usability of interactive systems. Developed by John Brooke in 1986, the SUS provides a quick and effective way to assess a system's overall usability through a standardized ten-item questionnaire. Each item is rated on a five-point Likert scale, ranging from "Strongly Disagree" to "Strongly Agree," and the results are used to compute an overall usability score.

The SUS evaluates usability across key aspects such as:

- **Effectiveness**: How well users can achieve their goals using the system.
- **Efficiency**: The ease and speed with which users can complete tasks.
- **Satisfaction**: The subjective experience of users while interacting with the system.

In this study, the SUS was utilized to gauge the usability of the enhanced eShadow platform. Participants completed the SUS after their interactions with the system, allowing for a quantitative measure of usability. The SUS's simplicity and general applicability make it a valuable tool for comparative usability studies and a benchmark for assessing user satisfaction.

The SUS is particularly well-suited for quick evaluations and has been validated across a wide range of applications, ensuring the reliability of its results. For more information on the SUS and its scoring methodology, refer to the original work by John Brooke or other resources available online.

### 7.4.4 SUS Results

This section is divided into three main parts. The first part discusses the results from the SUS evaluation conducted during the workshops at schools as part of the Chania Film Festival (Experiment 1). The second part presents feedback gathered from participants of the

Science Day event (Experiment 2). Finally, the third part compares the findings from these two experiments

A total of **58** valid questionnaires were included in the final analysis.

#### 7.4.4.1 Experiment 1

The SUS evaluation for the Chania Film Festival workshops yielded the following results from **20 questionnaires**:

Table 7.4: SUS Evaluation Results for Chania Film Festival Workshops

| Dimension | Average | Standard Deviation |
|---|---|---|
| SUS Score | 74.88 | 16.25 |
| Usability Score | 75.25 | 14.64 |
| Learnability Score | 74.50 | 22.30 |

**Interpretation:** The SUS evaluation results indicate that the platform provides a fairly satisfactory user experience. Below is the detailed analysis:

- **SUS Score (74.88, 16.25):** The overall score is above the benchmark threshold of 68, indicating an above-average user experience. However, the relatively high standard deviation suggests variability in user feedback.
- **Usability Score (75.25, 14.64):** The usability score highlights that participants generally found the platform user-friendly, with moderate consistency in responses.
- **Learnability Score (74.50, 22.30):** While users found the platform fairly intuitive, the higher standard deviation suggests that some participants struggled more than others to learn and navigate the system.

**Benchmarking:** When compared to benchmarks, the SUS scores achieved in **Experiment 1** place the system in the **72.8th** Percentile ranking the eShadow LCE system in the top 30% of the evaluated systems. Even though above average, the results indicate that there is still room for improvement in the system's usability.

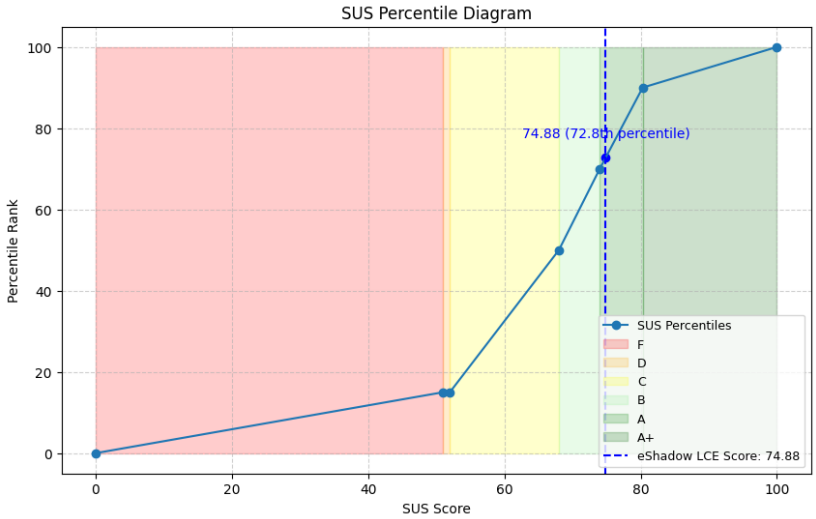



Figure 7.20: Sus Percentile Diagram

#### 7.4.4.2 Experiment 2

The SUS evaluation for the Science Day event produced the following results from **38 questionnaires**:

Table 7.5: SUS Evaluation Results for Science Day

| Dimension | Average | Standard Deviation |
|---|---|---|
| SUS Score | 82.89 | 12.74 |
| Usability Score | 83.03 | 15.27 |
| Learnability Score | 82.76 | 15.93 |

**Interpretation:** The Science Day evaluation highlights significant improvements compared to the earlier workshops. Below is the detailed analysis:

- **SUS Score (82.89, 12.74):** The score indicates a strong user experience, with the lower standard deviation reflecting more consistent feedback.
- **Usability Score (83.03, 15.27):** Participants rated the platform highly for usability, showing that it met or exceeded expectations for ease of use.
- **Learnability Score (82.76, 15.93):** Users found the platform intuitive and easy to learn, with relatively consistent responses across the group.

**Benchmarking:** When compared to benchmarks, the SUS scores achieved in **Experiment 2** place the system in the **91.3th** Percentile ranking the eShadow LCE system in the top 9% of the evaluated systems. The high learnability score, in particular, underlines the success of the intuitive design choices aimed at minimizing the learning curve for users.

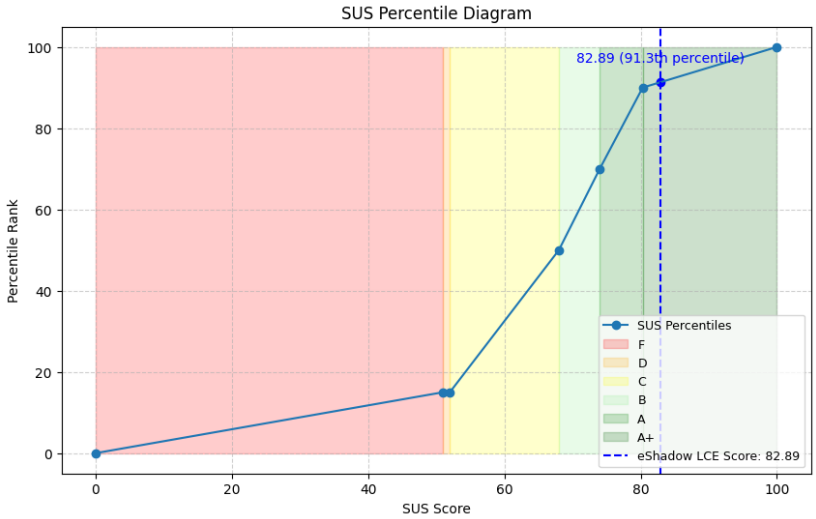



Figure 7.21: Sus Percentile Diagram

#### 7.4.4.3 Comparison of Results

The following table compares the results from Experiment 1 and 2:

Table 7.6: Comparison of SUS Evaluation Results

| **Dimension** | **Experiment 1** | | | **Experiment 2** | | |
|---|---|---|---|---|---|---|
| | **Average** | **SD** | **n** | **Average** | **SD** | **n** |
| SUS Score | 74.88 | 16.25 | 20 | 82.89 | 12.74 | 38 |
| Usability Score | 75.25 | 14.64 | 20 | 83.03 | 15.27 | 38 |
| Learnability Score | 74.50 | 22.30 | 20 | 82.76 | 15.93 | 38 |

**Interpretation:** The comparison reveals clear improvements in the Experiment 2 results:

- **SUS Score:** The Science Day event achieved a higher average score of **82.89** with a lower standard deviation of **12.74**, indicating a more uniformly positive user experience compared to the Chania Film Festival workshops, which scored an average of **74.88** with a standard deviation of **16.25**.
- **Usability and Learnability:** Both usability and learnability scores showed significant improvement in the Science Day results.This progress stems from meticulous observation and participant feedback from Experiment 1, combined with iterative refinements informed by comprehensive documentation of observations and challenges encountered during the Chania Film Festival workshops.

### 7.4.5 Smiley Face Test

The Smiley Face Test is a simple yet effective method for gathering quick feedback from users, often used in user experience (UX) research, especially with younger audiences or in

informal settings. This test presents participants with a range of smiley faces, typically varying from very happy to very sad, representing a spectrum of emotions or satisfaction levels. Users select the face that best reflects their feelings about a product, feature, or experience.

The simplicity of this test makes it highly accessible, requiring no prior training or technical knowledge. It is particularly useful in contexts where participants may find traditional questionnaires too complex or time-consuming. By relying on universally recognized emotional expressions, the test overcomes potential language barriers, making it effective in diverse user groups.

In the context of the eShadow platform, the Smiley Face Test can be used to evaluate immediate reactions to the platform's usability, design, or specific features. For example, students or educators could quickly express their satisfaction with the puppet creation process or the platform's collaborative tools by selecting a corresponding smiley face. This method provides valuable qualitative data that complements more structured evaluation tools like the System Usability Scale (SUS) or User Experience Questionnaire (UEQ).

While the Smiley Face Test is straightforward, its results are subjective and should be interpreted alongside other quantitative and qualitative metrics to gain a holistic understanding of user experience. Its primary advantage lies in its ability to capture raw, emotional feedback in a manner that is both intuitive and engaging for users.

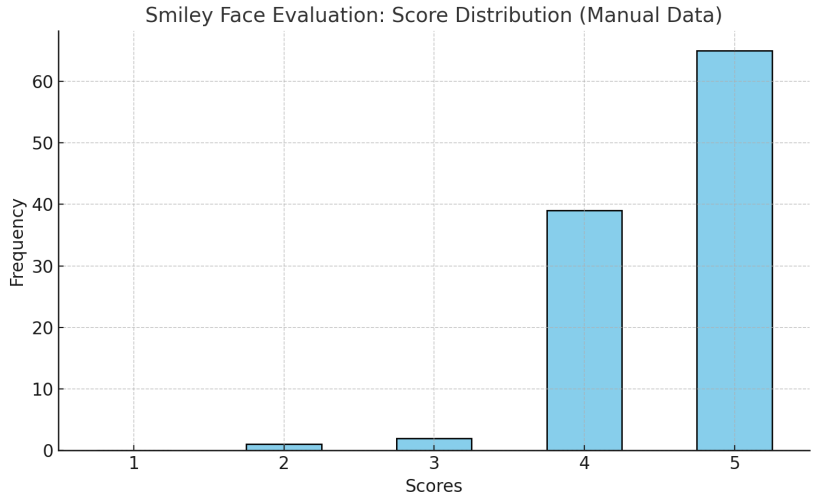



Figure 7.22: Smiley Vote Histogram

#### 7.4.5.1 Results

The analysis of the filtered scores is summarized as follows:

- **Number of Responses:** 108
- **Average Score:** 4.56
- **Standard Deviation:** 0.58

- **Median Score:** 5.00

#### 7.4.5.2 Interpretation

- The **mean score of 4.56** indicates that the majority of participants had a positive experience.
- The **standard deviation of 0.58** shows low variability, suggesting a consistent level of satisfaction across users.
- The minimum score of 2.00 highlights some dissatisfaction, but the maximum score of 5.00 reflects high levels of satisfaction among most users.

#### 7.4.5.3 Conclusion

The Smiley Face evaluation demonstrates strong overall satisfaction with the platform. While most users provided high ratings, the presence of a few lower scores suggests areas for potential improvement to enhance the user experience further.

### 7.4.6 Reflections on the Overall Results

The iterative process employed throughout the evaluation and between experiments has proven effective, allowing for the identification and resolution of key issues over time. Both the SUS and UEQ questionnaire results demonstrate clear improvements, aligning to form a cohesive and positive picture of the platform's usability and user experience. Additionally, the Smiley Face Test further reinforces this positive perception, solidifying the application as a pleasant and intuitive tool.

A minor limitation arose during the second UEQ experiment, where only 18 questionnaires were collected. This was due to challenges faced by younger participants in completing the UEQ questionnaire, which led to inconsistencies in the responses. These inconsistencies were identified and excluded through the UEQ sheet's built-in validation algorithm. To ensure the quality and accuracy of the final data, the second experiment's UEQ questionnaires were distributed exclusively to adults who had sufficient time to interact with the system.

Despite the smaller sample size in the second UEQ experiment, the upward trend observed in the results aligns closely with the SUS scores, suggesting that the findings are both reliable and accurate. Taken together, all results point to the application being not only highly usable but also enjoyable and engaging for its users.

# 8. Conclusions and Future Work

This thesis aimed to enhance the functionality and user experience of the eShadow platform by addressing its primary limitations in puppet creation and collaborative usability. By leveraging advanced technologies and methodologies, the work sought to bridge traditional shadow theater with modern digital solutions, fostering creativity, accessibility, and collaboration. The key outcomes and potential future directions are summarized below.

## 8.1 Summary of Contributions

The following contributions were achieved during the course of this work:

- **Streamlined Puppet Creation Process:** By integrating machine learning algorithms and generative AI techniques along with creative design decisions, the platform now enables users to create digital puppets from real-world objects and drawings effectively and efficiently. The process accommodates diverse inputs, including photos, sketches, and more complex artifacts, with minimal technical expertise required.
- **Collaborative Puppet Manipulation:** The introduction of real-time collaborative features allows multiple users to manipulate puppets simultaneously through their smartphones. This functionality enhances the platform's suitability for educational and creative workshops, fostering teamwork and engagement.
- **AI-Driven Enhancements:** The integration of advanced AI models, such as Stable Diffusion and ControlNet, enabled the automatic stylization of puppets, preserving the aesthetic of traditional shadow theater while accommodating user creativity.
- **Improved Accessibility and Usability:** A user-centered design approach, incorporating heuristic evaluations and iterative feedback, resulted in a platform that is intuitive, responsive, and accessible to both novice and experienced users.
- **Robust System Architecture:** The system's modular architecture, combining local server functionality with mobile app integration, ensures scalability, reliability, and seamless communication across devices.

## 8.2 Evaluation and Impact

The enhanced eShadow platform was evaluated across multiple user scenarios, including workshops, educational settings, and public events. The following outcomes were observed:

- Positive User Feedback: Participants consistently praised the platform's ease of use, engaging features, and creative potential, particularly in its ability to facilitate real-time collaboration and foster creativity in both individual and group settings.
- High Attractiveness and Stimulation Scores: User Experience Questionnaire (UEQ) results from both experiments showcased strong scores in **Attractiveness (2.60)** and **Stimulation (2.48)**, emphasizing the platform' s visual appeal, engaging nature, and ability to captivate users.
- Marked Improvements Across Dimensions: Compared to earlier evaluations, the iterative refinements between experiments resulted in significant improvements across multiple dimensions, particularly in **Efficiency (1.92)** and **Dependability (1.78)**, which highlight the platform's improved reliability and ease of use.
- Pragmatic and Hedonic Quality Balance: The platform achieved a commendable balance between **Pragmatic Quality (Efficiency and Dependability)** and **Hedonic Quality (Attractiveness and Stimulation)**, ensuring it supports users in achieving their goals while providing an enjoyable and motivating experience.
- Effective Implementation in Educational Contexts: The platform's success in engaging participants of various ages and backgrounds was evident during events such as the TUC Science Day and Chania Film Festival workshops. Despite challenges with younger participants completing the UEQ, data from adult participants with quality interaction time validated the platform's educational value and broad applicability.
- Validation Through Multiple Metrics: The alignment between UEQ, SUS, and Smiley Face Test results underscores the platform's usability and user satisfaction. The upward trends across all metrics suggest that the platform is not only usable but also enjoyable and well-suited for creative and educational environments.

These findings collectively highlight the eShadow platform as a versatile, user-friendly, and innovative tool, capable of delivering impactful experiences in diverse settings.

## 8.3 Limitations

Despite the significant advancements, the platform has certain limitations that warrant further attention:

- **Hardware Dependency:** The Generative AI component still faces deployment challenges due to the novelty of the technology and limited support for home-use applications. Additionally, most average users lack access to high-performance GPUs required for optimal functionality.
- **Scalability of Collaborative Features:** While real-time collaboration is functional, the current system supports a limited number of simultaneous users due to network and processing constraints.
- **AI Model Generalization:** The fine-tuned AI models may require further optimization to handle more diverse inputs and maintain consistent quality across all scenarios.

## 8.4 Future Work

Building upon the findings and outcomes of this thesis, several directions for future research and development are proposed:

- **Secure Server or Home Deployment for Style Transform:** Identifying a secure server or developing a method to deploy the style transformation component on home computers to make the feature more accessible to users without high-end hardware.
- **Intuitive Figure Controls:** Improving figure controls to allow users to intuitively define the control joints directly on the touch screen, enhancing the precision and ease of puppet manipulation.
- **Performance Optimization:** Enhancing the app's performance to ensure smooth operation, even on devices with moderate hardware capabilities.
- **Online Collaborative Play:** Developing the infrastructure for online collaborative play, enabling users from different locations to connect to the same server and participate in shared experiences. Currently, this feature is limited to local networks.
- **Multi-Platform Support:** Expanding support to include additional operating systems such as Linux and iOS, ensuring a broader user base can access the platform.
- **Bundling Main Server with eShadow Platform:** Finding a secure and efficient way to bundle the main server with the eShadow platform for seamless deployment and user convenience.
- **Support for Additional LoRAs:** Adding the capability to incorporate additional LoRAs (Low-Rank Adaptation models) trained on various art styles, allowing users to choose and apply diverse artistic aesthetics.

- **Streamlined Training for LoRAs:** Simplifying the training process for new LoRAs, enabling users to train and add their own models without requiring extensive technical expertise.
- **Multi-Language Support:** Support as many languages as possible.

In conclusion, the enhanced eShadow platform represents a significant step forward in merging traditional shadow theater with modern digital technologies. Its innovations in accessibility, creativity, and collaboration pave the way for broader adoption in educational and artistic domains, ensuring the preservation and evolution of this cultural art form in the digital age.

# 9. References

# Bibliography


[1] Aitrepreneur, *Stable diffusion lora training with kohya ss (step-by-step guide)*, Accessed: 2024-12-07, 2023. [Online]. Available: https://www.youtube.com/watch?v=70H03cv57-o&t=1113s.

[2] AUTOMATIC1111, *Stable diffusion webui*, Accessed: 2024-12-07, 2024. [Online]. Available: https://github.com/AUTOMATIC1111/stable-diffusion-webui.

[3] Baidu AI Studio, *Ai implements shadow puppetry, inheriting a vanishing art*, https://aistudio.baidu.com/projectdetail/764130, 2020.

[4] P. Bottoni, S. Faralli, A. Labella, A. Malizia, M. Pierro, and S. Ryu, "Copuppet: Collaborative interaction in virtual puppetry," in *Transdisciplinary Digital Art. Sound, Vision and the New Screen: Digital Art Weeks and Interactive Futures 2006/2007, Zurich, Switzerland and Victoria, BC, Canada. Selected Papers*, Springer, 2008, pp. 326–341.

[5] J. Brooke, "Sus: A quick and dirty usability scale," *Usability Evaluation in Industry*, 1996.

[6] Y.-T. Cheng, T. K. Shih, and C.-Y. Lin, "Create a puppet play and interative digital models with leap motion," in *2017 10th International Conference on Ubi-media Computing and Workshops (Ubi-Media)*, IEEE, 2017, pp. 1–6.

[7] M. Christoulakis, "Design, implementation and experimental evaluation of digital shadow theater eshadow," Accessed: 2024-12-06, M.S. thesis, Technical University of Crete, 2015. [Online]. Available: https://dias.library.tuc.gr/view/31882?locale=en.

[8] M. Christoulakis, A. Pitsiladis, A. Moraiti, N. Moumoutzis, and S. Christodoulakis, "Eshadow: A tool for digital storytelling based on traditional greek shadow theatre," in *workshop Proceedings of the 8th International Conference on the Foundations of Digital Games*, 2013.

[9] F. Community, *Image: Advanced image manipulation in flutter*, https://pub.dev/packages/image, Accessed: 2024-12-06.

[10] F. Community, *Provider: State management for flutter*, https://pub.dev/packages/provider, Accessed: 2024-12-06.

[11] S. A. CompVis, *Stable diffusion v1.5 - pruned ema-only model*, Accessed: 2024-12-07, 2022. [Online]. Available: https://huggingface.co/stable-diffusion-v1-5/stable-diffusion-v1-5/blob/main/v1-5-pruned-emaonly.safetensors.

[12] M. Corporation, *Onnx runtime*, Online, Available at https://onnxruntime.ai, 2024.

[13] Daniil Efremov, *Rembg: Background removal tool*, Accessed: 2024-11-22, 2024. [Online]. Available: https://github.com/danielgatis/rembg.

[14] B. Duncan, *Archive: Archive and unarchive files in flutter and dart*, https://pub.dev/packages/archive, Accessed: 2024-12-06.

[15] Facebook AI Research, *Pytorch*, Accessed: 2024-11-22, 2024. [Online]. Available: https://pytorch.org/.

[16] Figma, Inc., *Figma*, Accessed: 2024-12-09, 2024. [Online]. Available: https://www.figma.com.

[17] O. Foundation, *Open source computer vision library*, Online, Available at https://opencv.org, 2024.

[18] D. Gatis, *Rembg: A tool to remove images' background*, https://github.com/danielgatis/rembg, Accessed: YYYY-MM-DD, 2021.

[19] H. Goebel *et al.*, *Pyinstaller: Freeze python programs into stand-alone executables*, https://pyinstaller.org, Version 6.10.0, 2024.

[20] Google, *Dart programming language*, https://dart.dev, 2024.

[21] Google, *Flutter framework*, https://flutter.dev, 2024.

[22] U. Güdükbay, F. Erol, and N. Erdogan, "Tradition offers artistic possibilities for new media technologies: An animation system for shadow theatre," in *International Symposium on Electronic Art*, 2000, pp. 86–96.

[23] D. Hart, *Uuid: Universally unique id generation in flutter and dart*, https://pub.dev/packages/uuid, Accessed: 2024-12-06.

[24] A. Hesham, *Flutter_barcode_scanner: Barcode and qr code scanning in flutter*, https://pub.dev/packages/flutter_barcode_scanner, Accessed: 2024-12-06.

[25] E. Hu, Y. Shen, P. Wallis, *et al.*, *Lora: Low-rank adaptation of large language models*, Framework for fine-tuning large models, including in Unity integrations., 2021.

[26] E. J. Hu, Y. Shen, P. Wallis, *et al.*, "Lora: Low-rank adaptation of large language models," *arXiv preprint arXiv:2106.09685*, 2021.

[27] HungHD, *Image_cropper: Crop images in flutter*, https://pub.dev/packages/image_cropper, Accessed: 2024-12-06.

[28] JetBrains, *Pycharm: The python ide for professional developers*, Accessed: 2024-12-07, 2024. [Online]. Available: `https://www.jetbrains.com/pycharm/`.

[29] W. Jiang and C. Cao, "Reconstruction: A motion driven interactive artwork inspired by chinese shadow puppet," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 1441–1442.

[30] Kohya, *Kohya's stable diffusion scripts*, Accessed: 2024-11-22, 2024. [Online]. Available: `https://github.com/kohya-ss/sd-scripts`.

[31] M. Lin, Z. Hu, S. Liu, M. Wang, R. Hong, and S. Yan, "Eheritage of shadow puppetry: Creation and manipulation," in *Proceedings of the 21st ACM international conference on Multimedia*, 2013, pp. 183–192.

[32] G. Marin, F. Dominio, and P. Zanuttigh, "Hand gesture recognition with jointly calibrated leap motion and depth sensor," *Multimedia Tools and Applications*, vol. 75, pp. 14 991–15 015, 2016.

[33] Merjic, *Majicmix realistic 麦橘写实*, Accessed: 2024-12-07, 2023. [Online]. Available: `https://civitai.com/models/43331/majicmix-realistic`.

[34] R. Molich and J. Nielsen, "Improving a human-computer dialogue," *Communications of the ACM*, vol. 33, no. 3, pp. 338–348, 1990.

[35] A. Moraiti, "Σχεδίαση και υλοποίηση εφαρμογής για δημιουργία και επεξεργασία φιγούρων και σκηνικών ελληνικού θεάτρου σκιών," Accessed: 2024-12-16, M.S. thesis, Technical University of Crete, 2014. [Online]. Available: `https://dias.library.tuc.gr/view/23055`.

[36] A. Moraiti, N. Moumoutzis, M. Christoulakis, *et al.*, "Playful creation of digital stories with eshadow," in *2016 11th International Workshop on Semantic and Social Media Adaptation and Personalization (SMAP)*, IEEE, 2016, pp. 139–144.

[37] N. Moumoutzis, M. Christoulakis, S. Christodoulakis, and D. Paneva-Marinova, "Renovating the cultural heritage of traditional shadow theatre with eshadow: Design, implementation, evaluation and use in formal and informal learning," *Digital Presentation and Preservation of Cultural and Scientific Heritage*, vol. 8, pp. 51–70, 2018.

[38] N. Moumoutzis, M. Christoulakis, C. Xanthaki, *et al.*, "Eshadow+: Mixed reality storytelling inspired by traditional shadow theatre," in *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*, IEEE, 2022, pp. 95–100.

[39] N. Moumoutzis, N. Gioldasis, G. Anestis, M. Christoulakis, G. Stylianakis, and S. Christodoulakis, "Employing theatrical interactions and audience engagement to enable creative learning experiences in formal and informal learning: Enriching social and community theatre practices with digital technologies," in *Interactive Mobile Com-*

*munication Technologies and Learning: Proceedings of the 11th IMCL Conference*, Springer, 2018, pp. 142–154.

[40] N. Moumoutzis, A. Koukis, C. Xanthaki, *et al.*, "Epuppet: A mobile app to extend a digital storytelling platform with new capabilities," in *Interactive Mobile Communication, Technologies and Learning*, Springer, 2021, pp. 917–926.

[41] J. Nielsen, "Enhancing the explanatory power of usability heuristics," in *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, 1994, pp. 152–158.

[42] J. Nielsen and R. Molich, "Heuristic evaluation of user interfaces," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, 1990, pp. 249–256.

[43] NVIDIA Corporation, *Cuda toolkit*, Accessed: 2024-11-22, 2024. [Online]. Available: `https://developer.nvidia.com/cuda-zone`.

[44] NVIDIA Corporation, *Cuda toolkit documentation*, Available at: `https://developer.nvidia.com/cuda-toolkit`.

[45] OpenFlutter, *Flutter_image_compress: Image compression for flutter apps*, `https://pub.dev/packages/flutter_image_compress`, Accessed: 2024-12-06.

[46] Pallets Projects, *Flask: Web development, one drop at a time*, Available at: `https://flask.palletsprojects.com/`.

[47] M. Palma, *Smooth_page_indicator: Customizable page view indicators for flutter*, `https://pub.dev/packages/smooth_page_indicator`, Accessed: 2024-12-06.

[48] Pillow Contributors, *Pillow (pil fork)*, Accessed: 2024-11-22, 2024. [Online]. Available: `https://python-pillow.org/`.

[49] Python Software Foundation, *Python*, Accessed: 2024-11-22, 2024. [Online]. Available: `https://www.python.org/`.

[50] X. Qin, Z. Zhang, C. Huang, *et al.*, "$U^2 - net$: Going deeper with nested u-structure for salient object detection," *Pattern Recognition*, vol. 106, p. 107 404, 2020.

[51] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.

[52] M. Schrepp, A. Hinderks, and J. Thomaschewski, *User experience questionnaire (ueq)*, Accessed: 2024-12-04, 2023. [Online]. Available: `https://www.ueq-online.org/`.

[53] D. Team, *Http: A composable, future-based api for http requests*, `https://pub.dev/packages/http`, Accessed: 2024-12-06.

[54] F. Team, *Image_picker: Pick images from gallery or camera in flutter*, `https://pub.dev/packages/image_picker`, Accessed: 2024-12-06.

[55] F. Team, *Path_provider: Access filesystem paths in flutter*, https://pub.dev/packages/path_provider, Accessed: 2024-12-06.

[56] F. Team, *Shared_preferences: Persistent storage for key-value pairs in flutter*, https://pub.dev/packages/shared_preferences, Accessed: 2024-12-06.

[57] This is Athens. "Spathario museum of shadow theatre." (2024), [Online]. Available: https://www.thisisathens.org/museums/spathario-museum-shadow-theatre (visited on 11/18/2024).

[58] UNESCO. "Chinese shadow puppetry." (2024), [Online]. Available: https://ich.unesco.org/en/RL/chinese-shadow-puppetry-00421 (visited on 11/18/2024).

[59] Unity Technologies, *Unity*, Available at: https://unity.com/.

[60] Wikipedia Contributors. "Shadow play." (2024), [Online]. Available: https://en.wikipedia.org/wiki/Shadow_play (visited on 11/18/2024).

[61] Wikipedia Contributors. "Wayang kulit." (2024), [Online]. Available: https://en.wikipedia.org/wiki/Wayang_kulit (visited on 11/18/2024).

[62] Wikipedia contributors, *Kinect – Wikipedia, The Free Encyclopedia*, [Online; accessed 19-November-2024], 2024. [Online]. Available: https://en.wikipedia.org/wiki/Kinect.

[63] R. Wilinski, *Curved_navigation_bar: Beautiful and customizable curved navigation bars for flutter*, https://pub.dev/packages/curved_navigation_bar, Accessed: 2024-12-06.

[64] World Encyclopedia of Puppetry Arts. "Karaghiozis." (2012), [Online]. Available: https://wepa.unima.org/en/karaghiozis/ (visited on 11/18/2024).

[65] Z. Yao, S. Lyu, Y. Lu, *et al.*, "Shadowmaker: Sketch-based creation tool for digital shadow puppetry," in *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, 2024, pp. 1–5.

[66] L. Zhang and M. Agrawala, *Controlnet: Adding conditional control to text-to-image diffusion models - canny version*, Accessed: 2024-12-07, 2023. [Online]. Available: https://huggingface.co/lllyasviel/sd-controlnet-canny.

[67] L. Zhang and M. Agrawala, *Controlnet: Adding conditional control to text-to-image diffusion models - softedge version*, Accessed: 2024-12-07, 2023. [Online]. Available: https://huggingface.co/lllyasviel/control_v11p_sd15_softedge.

[68] L. Zhang, A. Rao, and M. Agrawala, "Adding conditional control to text-to-image diffusion models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 3836–3847.

# A. Supplementary Information

ΕΡΩΤΗΜΑΤΟΛΟΓΙΟ ΑΞΙΟΛΟΓΗΣΗΣ
ΤΗΣ ΕΦΑΡΜΟΓΗΣ **eShadow Companion / eShadow LCE (Local Collaboration edition)**

Επιλέξτε για κάθε πρόταση παρακάτω το βαθμό στον οποίο διαφωνείτε ή συμφωνείτε ως εξής:
1 – Διαφωνώ
2 – Μάλλον διαφωνώ
3 – Ουδέτερη γνώμη
4 – Μάλλον συμφωνώ
5 – Συμφωνώ
Βεβαιωθείτε ότι έχετε απαντήσει σε όλες τις ερωτήσεις. **Αν δεν ξέρετε τι να απαντήσετε επιλέξτε το 3.**

**Νομίζω ότι θα ήθελα να χρησιμοποιώ το eShadow Companion / eShadow LCE συχνά:**
Διαφωνώ - 1 2 3 4 5 - Συμφωνώ
**Βρήκα το eShadow Companion / eShadow LCE αδικαιολόγητα περίπλοκο:**
Διαφωνώ - 1 2 3 4 5 - Συμφωνώ
**Βρήκα το eShadow Companion / eShadow LCE εύκολο στη χρήση:**
Διαφωνώ - 1 2 3 4 5 - Συμφωνώ
**Νομίζω ότι θα χρειαστώ τεχνική βοήθεια για να χρησιμοποιήσω το eShadow Companion / eShadow LCE :**
Διαφωνώ - 1 2 3 4 5 - Συμφωνώ
**Βρήκα τις διάφορες λειτουργίες στο eShadow Companion / eShadow LCE καλά ολοκληρωμένες:**
Διαφωνώ - 1 2 3 4 5 - Συμφωνώ
**Νομίζω ότι υπήρχε μεγάλη ασυνέπεια στο eShadow Companion / eShadow LCE:**
Διαφωνώ - 1 2 3 4 5 - Συμφωνώ
**Φαντάζομαι ότι οι περισσότεροι άνθρωποι θα μάθουν να χρησιμοποιούν το eShadow Companion / eShadow LCE πολύ γρήγορα:**
Διαφωνώ - 1 2 3 4 5 - Συμφωνώ
**Βρήκα το eShadow Companion / eShadow LCE πολύ περίπλοκο/δύσκολο στη χρήση:**
Διαφωνώ - 1 2 3 4 5 - Συμφωνώ
**Ένιωσα πολύ σίγουρος/η χρησιμοποιώντας το eShadow Companion / eShadow LCE :**
Διαφωνώ - 1 2 3 4 5 - Συμφωνώ
**Χρειάστηκε να μάθω πολλά πράγματα πριν μπορέσω να ξεκινήσω με το eShadow Companion / eShadow LCE :**
Διαφωνώ - 1 2 3 4 5 - Συμφωνώ

Κυκλώστε τη φατσούλα που αντιπροσωπεύει την εμπειρία που είχατε με την εφαρμογή : eShadow Companion / eShadow LCE (Local Collaboration edition)

Παρακαλείστε να αξιολογήσει το προϊόν τώρα , σημειώνοντας ένα κύκλο ανά γραμμή .

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | | |
|---|---|---|---|---|---|---|---|---|---|
| ενοχλητικό | ○ | ○ | ○ | ○ | ○ | ○ | ○ | απολαυστικό | 1 |
| δυσνόητο | ○ | ○ | ○ | ○ | ○ | ○ | ○ | κατανοητό | 2 |
| δημιουργικό | ○ | ○ | ○ | ○ | ○ | ○ | ○ | αναποτελεσματικό | 3 |
| εύκολο στη μάθηση | ○ | ○ | ○ | ○ | ○ | ○ | ○ | δύσκολο στη μάθηση | 4 |
| πολύτιμο | ○ | ○ | ○ | ○ | ○ | ○ | ○ | υποδεέστερο | 5 |
| βαρετό | ○ | ○ | ○ | ○ | ○ | ○ | ○ | συναρπαστικό | 6 |
| αδιάφορο | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ενδιαφέρον | 7 |
| απρόβλεπτο | ○ | ○ | ○ | ○ | ○ | ○ | ○ | προβλέψιμο | 8 |
| γρήγορο | ○ | ○ | ○ | ○ | ○ | ○ | ○ | αργό | 9 |
| εφευρετικό | ○ | ○ | ○ | ○ | ○ | ○ | ○ | συμβατικό | 10 |
| παρελκυστικό | ○ | ○ | ○ | ○ | ○ | ○ | ○ | υποστηρικτικό | 11 |
| καλό | ○ | ○ | ○ | ○ | ○ | ○ | ○ | κακό | 12 |
| περίπλοκο | ○ | ○ | ○ | ○ | ○ | ○ | ○ | εύκολο | 13 |
| αντιπαθητικό | ○ | ○ | ○ | ○ | ○ | ○ | ○ | συμπαθητικό | 14 |
| συνηθισμένο | ○ | ○ | ○ | ○ | ○ | ○ | ○ | πρωτοπόρο | 15 |
| δυσάρεστο | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ευχάριστο | 16 |
| ασφαλές | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ανασφαλές | 17 |
| ενθαρρυντικό | ○ | ○ | ○ | ○ | ○ | ○ | ○ | αποθαρρυντικό | 18 |
| ανταποκρίνεται στις προσδοκίες | ○ | ○ | ○ | ○ | ○ | ○ | ○ | δεν ανταποκρίνεται στις προσδοκίες | 19 |
| ανεπαρκές | ○ | ○ | ○ | ○ | ○ | ○ | ○ | επαρκές | 20 |
| σαφές | ○ | ○ | ○ | ○ | ○ | ○ | ○ | μπερδεμένο | 21 |
| μη πρακτικό | ○ | ○ | ○ | ○ | ○ | ○ | ○ | πρακτικό | 22 |
| οργανωμένο | ○ | ○ | ○ | ○ | ○ | ○ | ○ | ανοργάνωτο | 23 |
| ελκυστικό | ○ | ○ | ○ | ○ | ○ | ○ | ○ | απωθητικό | 24 |
| φιλικό | ○ | ○ | ○ | ○ | ○ | ○ | ○ | εχθρικό | 25 |
| συντηρητικό | ○ | ○ | ○ | ○ | ○ | ○ | ○ | καινοτόμο | 26 |

Κυκλώστε τη φατσούλα που αντιπροσωπεύει την εμπειρία που είχατε με την εφαρμογή : eShadow Companion / eShadow LCE (Local Collaboration edition)