7CCSMPRJ
MSc. Final Project Report

Innovaint: An Accessible Curation Space for Expression

Anon author

Abstract

Aphasia is a language processing disorder, which creates barriers to effective communication. This hinders the everyday interactions and social participation of those with aphasia. This project aims to assist people with aphasia to express themselves using overlap between communication and art therapy. Innovaint is an augmented reality painting application that conveys their message through visual arts using artistic style transfer techniques. It explores the medium of augmented environments over traditional physical therapy. A comprehensive review of design aspects employed has been presented. Results from expert proxy evaluations of the application have been reported, which confirm the potential benefit to and usability of the application by people with aphasia.

Keywords - aphasia, expression, communication therapy, art therapy, Augmented Reality, Artistic Style Transfer

Originality Avowal

I verify that I am the sole author of this report, except where explicitly stated to the contrary. I grant the right to King's College London to make paper and electronic copies of the submitted work for purposes of marking, plagiarism detection and archival, and to upload a copy of the work to Turnitin or another trusted plagiarism detection service. I confirm this report does not exceed 15,000 words.

Anon

Acknowledgements

Anon

Contents

List of Figures

List of Tables

Listings

Nomenclature

3D Three-Dimensional

AAC Augmentative and Alternative Communication

AR Augmented Reality

AST Artistic Style Transfer

CNN Convolutional Neural Network

CPU Central Processing Unit

DSP Digital Signal Processing

EU European Union

GPU Graphics Processing Unit

I/O Input/Output

NLP Natural Language Processing

OS Operating System

PWA People with Aphasia

RAM Random Access Memory

RGD Red, Green, Blue

SDK Software Development Kit

SLT Speech and Language Therapist

TTS Text-To-Speech

UI User Interface

Chapter 1

Introduction

1.1 Motivation

Aphasia is a disorder that arises from damages to areas of the brain responsible for producing and processing language. This damage deteriorates the ability to speak, read, write, and understand language [4]. The intensity of aphasia could range from mild to very severe, making it impossible to communicate in severe cases. Approximately 180,000 new cases of aphasia emerge each year in USA alone [5]. Hence, it is crucial to explore technology to aid people with aphasia to improve their well being.

Aphasia is a language processing disorder that disrupts the ability to access ideas and thoughts through language, not the ideas and thoughts themselves [5]. The loss of communication, the most fundamental element of social engagement, is sometimes the most distressing transition in one's life [6]. Therefore, there is a need to study alternative ways of communication for people with aphasia to help them express themselves.

Aphasia therapy involves helping people in enhancing their language abilities to help them communicate better or providing them alternative communication forms such as using gestures, visual arts, or electronic mediums [6]. Art is a visual form of communication that people with aphasia can use. This project aims to aid them in communication by using visual art as a form of expression.

1.2 Aims and Objectives

This project aims to create a mobile Augmented Reality (AR) painting application for people with aphasia to aid communication therapy using art expression. This provides them with an alternative way of communication through visual arts. The application would allow users to create paintings by placing virtual objects in the real world. Users will be able to curate various scenes as per desire, capture the augmented landscape, and apply various painting styles to generate paintings.

The objectives of this project are to

- Acquire relevant virtual objects
- Design the AR environment
- Design and Develop the AR interface
- Acquire relevant style filter images
- Evaluate and integrate the Artistic Style Transfer model
- Develop an interface to access the AST model
- Integrate the AR interface with the AST interface
- Design the flutter user interface
- Test and Deploy the application
- Evaluate the application through proxies

1.3 Challenges

Due to the time constraint of this project, the application could not be tested on actual people with aphasia. As this would have required a higher risk clearance than the current one, it was not possible to get approval for that in the limited time. Moreover, because of the covid situation, face-to-face focus groups and interviews were not possible with the proxies.

The application uses a few plugins to include some functionality. An essential plugin integrates the augmented reality interface with flutter, which is an extension of using "Unity As A Library" [7]. Hence, the application had to be developed considering the support and compatibility of these plugins.

1.4 Organization

The content structure of the report is detailed below.

The Background chapter reviews the existing work in this domain and includes background theories required to implement the project.

The Designs and Specifications chapter explains the design process employed through the project, details the design considerations and lists the functional and non-functional requirements for developing the application.

The Implementation and Methodology chapter defines the methodology followed, outlines the modules and provides the technical details for implementing each module.

The Proxy Evaluations chapter details the formative and final evaluations by the proxies and analysis and discussion of these evaluations.

The Innovaint Overview chapter walks through all the primary and additional features of the application. It also provides displays screenshots of using the application.

The Technical Evaluation chapter includes the results of the functional and performance testing of the application.

The Legal, Social, Ethical and Professional Issues chapter states the considerations involved and catered to throughout the project.

The Conclusion chapter concludes the report, describes the limitations of the project and recommends possible extensions as future work.

Chapter 2

Background

2.1 People with Apahasia

About 2 million people in the US suffer from Aphasia [8]. Aphasia is a language and communication impairment most caused by a stroke or other types of brain injury which damages portions of the brain responsible for language production and comprehension. The capability to understand speech, speak, read, write is affected. The disorder may primarily affect a single aspect of communication or multiple aspects together. However, the intelligence of affected people remains intact [8].

2.1.1 Functional Applications

Recovery from aphasia could possibly take years or decades. Improvement is a slow process in which support needs to be provided to the affected individuals and their families. Support takes the form of making them understand the disorder, in addition to teaching alternative communication mechanisms. The lack of communication disrupts the social participation of these people by making two-way communication difficult. Hence, families and caregivers need to be taught how to communicate with them [9].

To help the therapist's technology has played an important role. Many speaking-skills improving, interactive multi-media pro-games have been developed [10]. Smalltalk is also an application for people with aphasia, specifically targeting those who have speaking issues. It helps them by providing commonly used words and phrases and also enables them to practice frequently used words [11]. Theraphasia is a mobile-based application aimed at the rehabilitation of speech, in which therapists can design short games for the patient with the

help of pictures, text-to-speech, and word and sentence comprehension, and the ability to track the user's performance remotely [12]. Similarly, the StepByStep application also uses language therapy to aid people with aphasia (StepByStep Aphasia Therapy, 2021).

2.1.2 Creative Applications

Moreover, there have been multiple attempts to provide people with aphasia creative tools to facilitate their creativity and explore their creative side, considering the disorder does not affect a person's intellect. The Make Write application provides users the ability to write creative content by introducing features like rearranging text taken from paragraphs to form sentences, editing the writing, and sharing it on social media [13]. Another similar application is the Comic Spin which enables users to create comics by combining content information using images and text, which helps to explore the creative ability of people with aphasia [14]. Neate et al. proposed a tangible platform for people with aphasia called CreaTable, which allows them to create and interact with digital content [15].

There is an immense need to communicate with people with aphasia and enable them to communicate back. Numerous applications have been developed to facilitate people with aphasia in language therapy and functional activities. Although there has been some development in creative applications, the space of providing them platforms to explore or aid their creativity is still new and poses gaps to be filled. This project tries to bridge this gap by proposing a creative platform while aiding communication at the same time.

2.1.3 People with Aphasia and Therapy

Total communication-based therapy comes under the branch of speech therapy for people with aphasia which helps them express themselves and their feelings using alternative ways of communication. This is mostly used in combination with their existing language abilities [16]. There have been attempts to explore the possible alternative ways of communication such as humor [17], APPUTE intervention [18], PACE therapy [19] and art therapy [20].

APPUTE intervention is a technique aimed at people with severe aphasia. It involves equally including the partner and caregivers in therapy sessions in addition to the person with aphasia. This objective is to explore and derive communication techniques between them that would help convey everyday messages. The study showed promising results by improving both the communication skills between them and the individual linguistic skills of the person with aphasia. [18]. PACE therapy relies on combinations of multiple forms of communication to

get the message across, ranging from speaking, writing, drawing, using gestures, and even alternative communication devices [19].

Finally, the medium of art therapy has shown potential on being used as a form of expression [21]. According to the American Art Therapy Association, art therapy employs psychotherapeutic techniques combined with the creative process to improve the well-being of an individual [22]. In the context of aphasia, art therapy could be used to ease communication by using visual arts, music, or literature as a form of expression. Art therapy has also shown improvement in communication skills of chronic stroke survivors [23]. Drawing is frequently viewed as a form of aesthetic expression, it serves a very practical purpose; it enables us to communicate thoughts that cannot be effectively expressed through words [24].

This study focuses on using visual arts as a medium in communication therapy to aid people with aphasia in expressing themselves. It employs augmented space and virtual objects instead of the traditional physical forms of art therapy. Also, generating unique art pieces with minimal input while sustaining the goal of expression as in art therapy sessions with physical mediums. This enhances ease of use and creates endless possibilities of expression, which might not have been practical in physical environments.

2.2 Augmented Reality

The competency to convert computer graphics into the real world is called Augmented Reality (AR). Augmented Reality allows the user to see the real world and the virtual world simultaneously and interact with it and also allows attaching virtual objects to real locations and objects. There are currently 810 million active mobile AR users [25], and it is projected to increase to 3.5 billion users by 2022 worldwide [26].

The first Augmented Reality based system was created in Sutherland in 1968 [27]. In 2000, Bruce Thomas was the first to create the AR Quake outdoor smartphone AR game, which he presented at the International Symposium on Wearable Computers [28]. In 2005, a camera device based on augmented reality technologies was created that could observe the physical world in real-time and compare objects, and environment locations [29]. Many AR-based mobile applications started to appear in the following years.

2.2.1 Popular Augmented Reality Applications

Houzz is the most popular app for home interior design which also provides the user to purchase these products. The "View in My Space" feature uses Augmented Reality to position items into a picture of the user's home, which is then rendered in 3D for a lifelike image [30]. Another AR app for iPhone and Android that is based on home decor is IKEA Location. The Swedish furniture store allows customers to try out their purchases in their own houses. This application considers the larger picture, examining the whole floor plan to determine which objects best suit where. The ability to see various colors and drag-and-drop features almost takes the joy out of the IKEA experience [31].

2.2.2 Augmented Reality Painting Applications

World Brush is a virtual reality technology that helps users make site-specific art anonymously anywhere in the world. Within the app, users can find nearby art or explore inspirational work created in other locations. World Brush is a fun and innovative way to communicate with others in augmented reality. Users are invited to provide suggestions on other people's work, and to up-vote excellent work [32]. Figment AR is an augmented reality technology that allows users to create creative scenarios in their environment. Users will surround themselves with emojis, animals, and other fun items, build "portals" to travel to different dimensions, and incorporate effects like frost, fireworks, and more. To bring these figments of the imagination to fruition, users could share these moments with peers [33]. Holo is an AR application that is the closest to reality because it helps users to place holograms of actual people and animals anywhere in the world. Until now, AR has mainly consisted of simulated 3D models or photographs, but Holo has changed that [34].

2.2.3 Augmented Reality Software Development Kits

The most popular Software Development Kits (SDK) for developing Augmented Reality applications are Google's ARCore, Apple's ARKit, Wikitude, and Vuforia [35] [36].

ARCore supports advanced AR features, in addition to the basic ones. It provides motion tracking, environment understanding, and light estimation using the phone's camera. ARCore uses ray detection with detected planes and feature points relevant to the user's position and orientation in the real world. This SDK is compatible with devices running Android 7.0 or later and iOS 11.0 or later [35].

Apple's ARKit SDK includes 3D scanning and world tracking features in addition to the

		ARCore	ARKit	Wikitude	Vuforia
OS (min)	**Android**	7.0	-	6.0	6.0
	iOS	11.0	11.0	12.0	11.0
Features		Points, Plane Detection, Pose, Light estimation, Anchors, Trackables, Image and Face Tracking, Object Occlusion, Cloud Anchors	Plane Detection, Pose, Light estimation, Anchors, Trackables, Face Tracking, 3D Scanning, AR world map, Motion Capture, People Occlusion	Trackables, Geolocation, Cloud Recognition, Distance-based Scaling	3D Scanning, Predefined Trackables, Object Recognition

Table 2.1: Comparison of AR Software Development Kits (based on [35] [36])

features provided by ARCore. Moreover, it allows people occlusion and motion capture for limited devices. ARKit is built for developing applications for the iOS operating system only and supports devices running iOS versions greater than 11.0 [35] [36].

Wikitude is a relatively recent open-source SDK for the development of AR applications. It supports both the Android and iOS operating systems, with versions of Android greater than 6.0 and iOS versions greater than 12.0. Wikitude features geolocation, cloud recognition, and distance-based scaling features in addition to a range of tracking methods and technologies [36].

Finally, the Vuforia SDK was essentially designed for object and image tracking in AR development. It has key features enhancing object recognition and 3D modeling capabilities for the applications. The Vuforia SDK could also be used in fusion with ARCore and ARKit development kits. It supports development for Android versions greater than 6.0 and iOS versions greater than 11.0 [35] [36]. Table 2.1 summarizes the contrast between these SDKs.

With the introduction of Augmented Reality, a pattern has emerged of using advanced digital VR/AR technologies to develop forgiving experiences for visualizing challenging working scenarios, gaining risk-prevention skills, and receiving training. Furthermore, the use of AR immensely enhances the spectrum for creative and artistic experiences.

This project showcases a novel approach to using AR experiences in communication therapy employing art therapy. Exploring augmented environments to aid people with aphasia to express themselves has not been touched upon till yet. Augmented Reality provides people with aphasia the ability to create and curate content using a mix of the real and virtual environment instead of being confined to the world of fiction or reality only. This facilitates their creativity by providing an entirely new perspective on the traditional forms of art.

2.3 Artistic Style Transfer

Artistic style transfer (AST) is a promising tool for transferring the image content in the style of another by using a computer vision technique. AST is an illustration of image stylization, a technique for processing and manipulating vision data, that has been analyzed over decades within the field of non-photorealistic rendering. Figure 2.1 shows an example of an image style transfer. AST is used in various fields, including video and photo editors, commercial art, virtual reality, gaming, and artist-community engagement.

(a) Content Image (b) Style Image (c) Stylized Image

Figure 2.1: Artistic Style Transfer Example

2.3.1 Artistic Style Transfer Algorithms

Gatys et al. published the first study on artistic style transfer in 2015 [37]. The style transfer technique is built on a deep neural network, which uses neural representations to separate and recombine the content and style of random images, using a neural algorithm for producing artistic images. The technique presented in this study was based on using a Convolutional Neural Network (CNN) to create aesthetic pieces of art using the semantics of one image and the style of another. Neural Style Transfer algorithms could be broadly categorized into Descriptive Neural Methods and Generative Neural Methods [38]. These are summarized below as explained in [38].

Descriptive Neural Methods

The Descriptive Neural Methods are based on image iteration, where pixels of the style image are iteratively updated using Backpropagation to minimize the total loss to match the content of the content image and the style of the style image. There have been various adoptions of this loss function which could be grouped into Maximum Mean Discrepancy (MMD) methods

and Markov Random Fields (MRF) methods. MMD methods use the MMD metric as the style loss for optimization, and it has been proved as an effective style transfer method in [39] based on matching the feature distribution of the style and content image. The MMD algorithm was first used by the original paper introducing neural style transfer [37]. Then, a study by Li and Wand [40] introduced the MRF methods based on image synthesis on an abstract level compared to the per-pixel proposed by [37].

Generative Neural Methods

The Generative Neural Methods are based on model iteration, built to overcome the Descriptive Neural Methods' speed and computational cost issues while having some limitations. A feed-forward network is trained for each specific style image, and a separate network needs to be trained for each style. The model Is optimized using gradient descent. There have been numerous attempts to overcome the limitations. Johnson et al. [41] have proposed an algorithm that is three orders of magnitude faster than the original style transfer algorithm [37]. Moreover, another study proposes a more generalized network by training the model with multiple style images simultaneously [42].

The Magenta [43] model used as part of this project is based on an algorithm by Ghiasi et al. [44], which merges the neural artistic style transfer algorithms with fast style transfer networks achieving real-time arbitrary artistic style transfer. The proposed model is trained on approximately 80,000 painting images, making generalization to unseen images possible. Furthermore, the study successfully applies style transfer from unobserved paintings in real-time in a compact embedding space. As a result, the model included in the application performs fast on-device style transfer.

2.3.2 Web Applications

Some of the popular web-based applications that use Artistic Style Transfer are DeepArt, Pikazo, and Lucid. DeepArt is a web application that permits operators to construct distinctive artistic images through an algorithm to redesign the image by using the stylistic aspect of another image [45]. DeepArt uses a neural style transfer algorithm to generate an illustration of a user-provided image by applying the style of another image. Pikazo is also a Neural Style Transfer based algorithm that allows the user to paint an image in another style to create a masterpiece, resulting in sometimes surprising, beautiful, or disturbing artwork [46]. It is deemed as a collaboration among machines, humans, and the abstraction of art. Lucid is

a visual collaboration suite that provides a company or team the potential to imaging the forthcoming followed by building it. It offers connectedness and precision that facilitates users to dream, collaborate and build in real-time [47].

2.3.3 Mobile Applications

More recently, mobile applications featuring Image Style Transfer have gained popularity with applications like Prisma, Pikazo, and Artisto. Prisma's convolutional neural network and deep learning algorithms enable users to convert their pictures into paintings with styles of famous artists such as Picasso and Van Gough [48]. The Pikazo application goes a step forward by allowing users to choose styles from their own pictures rather than providing a finite set of styles. The application uses a mechanism that works by simulating the human visual cortex. It generates a new image based on the original photo provided by the user and a chosen style [46]. Artisto is a more recently developed application that allows Image Style Transfer to be applied to videos in addition to images. It enables the user to choose a style, and the application generates a video based on the chosen style [49].

This work uses Artistic Style Transfer to aid people with aphasia to communicate using various colors and style materials to express themselves. The technique of Artistic Style Transfer provides people with aphasia a platform to create unique paintings in various artistic styles through minimal input. This, in turn, allows them to convert their imagination to reality by exploring an innovative art form.

Chapter 3

Design & Specification

3.1 Design Process

Being an assistive application, the design was largely influenced by literature on how people with aphasia interact with technology and direct input from researchers working in this domain. The design process was iterative, where an initial prototype was built using knowledge from the background work - Iteration 1. The initial prototype was then presented to experts for feedback. The feedback was incorporated into the application to enhance usability for people with aphasia and add relevant features - Iteration 2. The final prototype produced was then presented for a final proxy evaluation. The design process is summarized in figure 3.1.

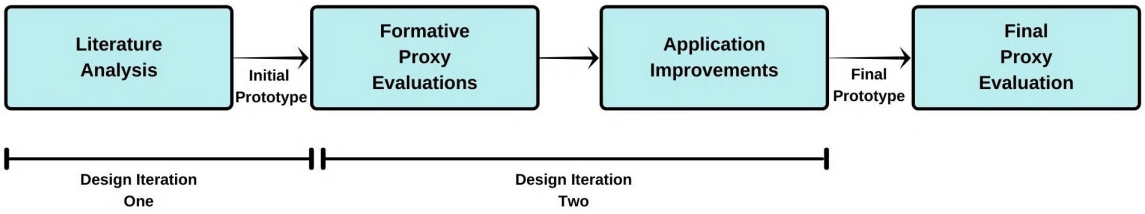


Figure 3.1: Design Process

3.1.1 Design Features from Literature

Numerous studies revolve around designing technology for people with aphasia, which helped form an understanding of designing the Innovaint application.

All user interface elements should be made large enough to support easy interaction, even when the non-dominant hand has been used. Moreover, none of the interactions should require very precise touch on the interface [13] [14]. The users should have control over the

level of customization in curating content, allowing them to choose the complexity and depth of creation [14]. The text used throughout the application should be plain and displayed in legible, moderate-sized fonts [14].

The application should have multi-modal support by communicating concepts in multiple ways, such as both text and audio. Also, the interface should be simple to use and formal with minimal steps to avoid complex interactions and design [14] [50]. Furthermore, it should support one-handed use to cater to people with a weak side because of brain injury [50].

According to the Language-Light UX guide for aphasia [51], the interface should be clutter-free with a focus on the main task. Moreover, familiar icons and everyday language should be used. Lastly, there should be straightforward navigation with consistent actions, and layouts [51].

3.1.2 Feedback by Proxies

The second iteration included involving proxies in the study for direct feedback on the design of the initial prototype. This is detailed in chapter 5 of this report.

3.2 Design Considerations

3.2.1 User Interface Design

Based on the detailed design considerations, the final user interface of the application was developed. A minimal design was employed throughout the application to avoid clutter and keep things simple to understand.

The application was designed to always support one-handed use, so it is usable for people who have a weaker side post-stroke. To facilitate this, buttons were made large and required to fill across the screen. Moreover, sliders were added for object interactions to support one-hand use over using gestures that require multi-touch. No such interactions which require a precise touch on the screen were added, considering people with aphasia struggle with fine motor skills.

To make it easier for users with reading difficulties, semi-bold sans-serif font was used throughout the application. In addition to text, icons were added to the buttons describing the specific button's task to support a multi-modal design.

Toggles to activate and deactivate interface elements in the augmented reality screen were included to allow the maximum space on the screen for scene curation. Similarly, the created scene and the final painting images were displayed to take the maximum space available. Lastly,

clear navigation icons were added to allow moving back and forth through the application easily.

3.2.2 Virtual 3D Objects

The application includes virtual three-dimensional objects that the user could play around with within the scene. Much thought has been put into selecting the objects' library to enable the users to curate scenes as a form of expression. The range of chosen objects has been spread across encompassing emotions such as happiness or anxiety, personal objects like a keyboard or a gardening trowel and ones aiding storytelling.

According to [52], colour, texture, form and movement can considerably influence feelings. Symmetrical and round objects are associated with positive emotions and calmness. On the other hand, objects with asymmetrical, sharp or angular shapes are associated with emotions like sadness, anxiety, excitement, or surprise [52] [53]. Considering these design aspects, objects of various shapes have been included in the object library, allowing the user to express an array of emotions.

Personal objects represent objects related to a particular aspect of one's personality, interests or emotional attachment. Lee explains that an object impacts how it connects to a specific piece of one's personality or memory. [52] Based on this, objects such as a cooking spoon and a camera have also been added to the application.

Finally, objects which help in storytelling have been included. Storytelling using cherished objects might create a familiar and neutral setting for socially isolated persons to engage in conversation and effective interaction with others [54]. In the past, studies [55] [56] have explored the potential of using technology to aid storytelling by people with aphasia and yielded promising results.

3.2.3 Painting Style Filters

The application has a library of painting styles to apply to the curated scene for style transfer. These filters have also been selected to allow for expression and communication. Mainly, the colours and materials of these painting styles have been considered. Hence, a range of colours and materials have been included. The combination of these enables expressing moods including but not limited to playful, cheerful, or tense.

Colours have been long known to be associated with emotions and expression. For instance, shades of blue have been connected to convey sadness, and yellow has been linked to joy, and red to anger or excitement [57]. Thus, the use of colours could express how one is feeling in a

particular moment. Based on this, painting styles that cover most of the colour spectrum have been included to facilitate expression.

The media used in art therapy highly influences a person's intent of expression. Materials range from communicating restricted to expressive emotions. For instance, fluid materials like watercolours or oil paints are explicit and provide more freedom, enhancing the emotional experiencing [58]. On the other hand, materials like pens and markers are structured and offer less control to the user, naturally expressing anxiety [58]. Based on the aforementioned, style images that have been created using a variety of media have been included in the styles' library.

Lastly, providing finite objects and styles would enhance creativity according to a finding by [14].

3.3 Feature Considerations

To ensure that very minimal input is needed to curate a scene, the choice of complexity and depth has been left up to the user. A basic user who might place only a single default object in the scene without any transformation to it would still get an aesthetically pleasing visual art piece at the end. The choice to save the painting has also been left out to the user's will, who can view the painting inside the application without saving or visiting the gallery.

For someone who wants to get more out of the application would be equally able to by having a choice to place as many objects as desired and also interact with them by transforming their size, rotation and position in the environment to curate a more complex scene, while capturing the process along the way as well. The application also provides a chance to change the image instead of using the last captured scene if the user is unhappy with it. Finally, it allows the user to save or share the generated painting, in addition to viewing it.

3.3.1 Text to Speech

As people with aphasia face language difficulties that may affect their reading abilities, a text-to-speech feature has been included in the application. The feature was identified as part of the proxy interviews, which suggested an audio feature would be beneficial. Hence, displayed text on every interface throughout the application can be read aloud. Moreover, the text is not automatically read out to avoid interfering with users who do not prefer this feature.

Technical Design

A Text-to-Speech system synthesizes speech from text. It uses principles from Natural Language Processing (NLP) and Digital Signal Processing (DSP) to analyze text and convert it to speech [59]. It is an assistive technology that reads out text, aiding people with visual impairments or reading disabilities.

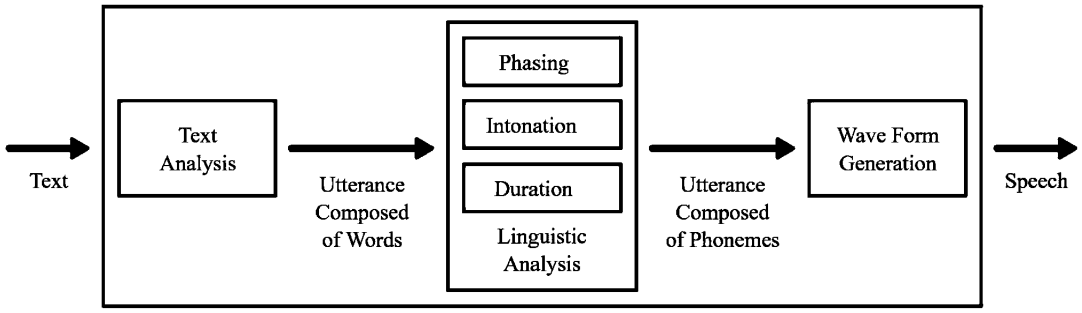


Figure 3.2: Text-To-Speech System Architecture [60]

Figure 3.2 shows the design of a typical text-to-speech system. A typical system is comprised of a front end, and back end [61]. The front end is responsible for preprocessing the text by converting it into a form of written words. This is followed by a grapheme-to-phoneme conversion which assigns phonetic transcriptions to each word and divides the text into prosodic units. This representation is then passed to the back end. The back end is the synthesizer that converts this linguistic representation to sound.

3.3.2 External Sharing

The user has been provided options to save the final curated painting to the gallery or share it across third-party applications. The rationale is that the user might want to revisit a curated piece to express something similar without going through the entire process again. Moreover, the sharing feature enables the user to express himself/herself to friends or family that are not physically nearby. This form of expression could simply be sharing an image of where the user is at that moment, but with an aesthetic touch to it, allowing users to turn bland surroundings into a playful one and vice versa.

3.3.3 Capturing the Augmented Scene

Photography is a preferable art medium for people with aphasia as it requires little motor control and is easily adaptable. It is also a familiar, simple, and easy to use medium of

expression, requiring minimal training for people who lack artistic talent or technical knowledge [20]. Photography has effectively been used to support communication, self-expression and empowerment of people with aphasia in the past [62]. Hence, the application included this feature to allow users to express themselves using features from their environment with minimal input.

3.4 Design Iterations

3.4.1 Iteration One

Augmented Reality Interface

The interface has a clean and minimal interface with clearly labelled, large-sized buttons which change colour on press. The major chunk of the screen is allocated to the environment to play around with and also includes toggles for hiding UI elements to make more room on the screen for object placement and interaction. Moreover, the object images are scaled when scrolling and a crosshair is provided to allow precise placement of objects. Figure 6.22 summarizes these design features.

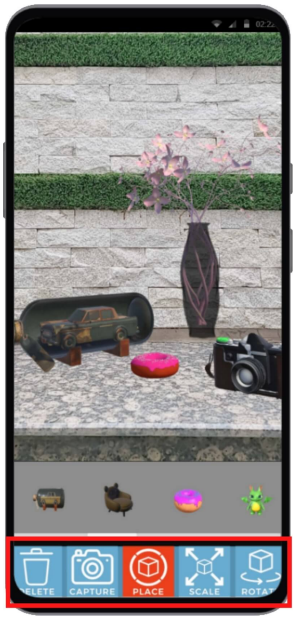
Artistic Style Transfer Interface

The AST interface is also a basic and simple interface with straightforward interaction. The buttons are filled across the screen for easy actions. The style filter images are clearly displayed. Lastly, the image area allocated the maximum space to present the curated scene and the generated painting. Figure 3.4iv shows these design features in the application.

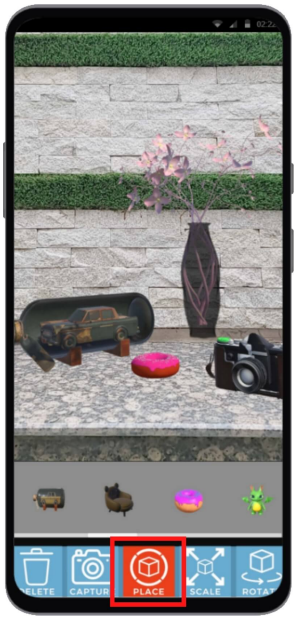
3.4.2 Iteration Two

Augmented Reality Interface

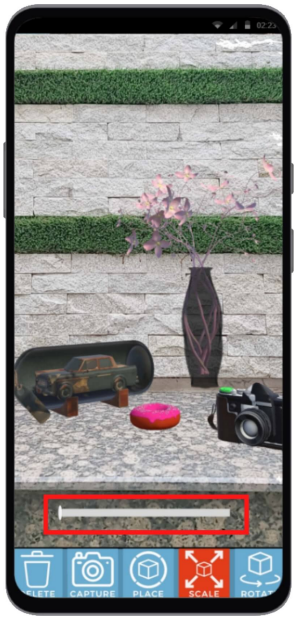
The buttons were changed to look consistent throughout the screen. Further, the scale and rotation sliders were colour-coded to highlight the slide function. The knob size of the sliders was increased to make it easier to slider. Also, the respective icon was added to reinforce the purpose of the slider. Lastly, speaker buttons were included and attached to each action button for enabling text-to-speech functionality. Figure 3.5 presents these improvements.



(i) Large buttons for easy interaction with icons and text to explain each action

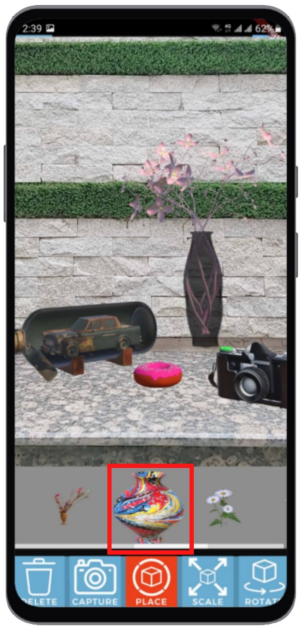


(ii) Button color change on-click to provide feedback for a better user experience

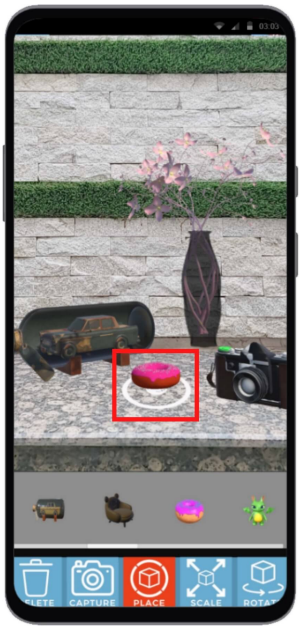


(iii) Slider to support one-handed use for people with a weaker side post stroke

(iv) Toggle-able elements to provide large area for curating scenes

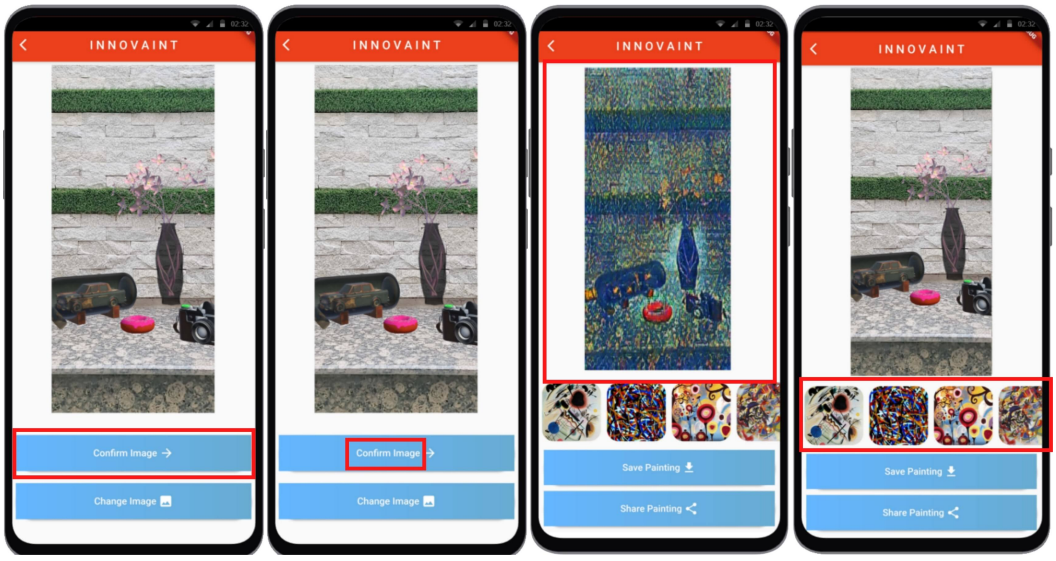


(v) Object image zoomed in when user scrolls over it to provide a clearer image



(vi) Crosshair for precise placement of objects in the environment

Figure 3.3: AR Interface - Iteration One



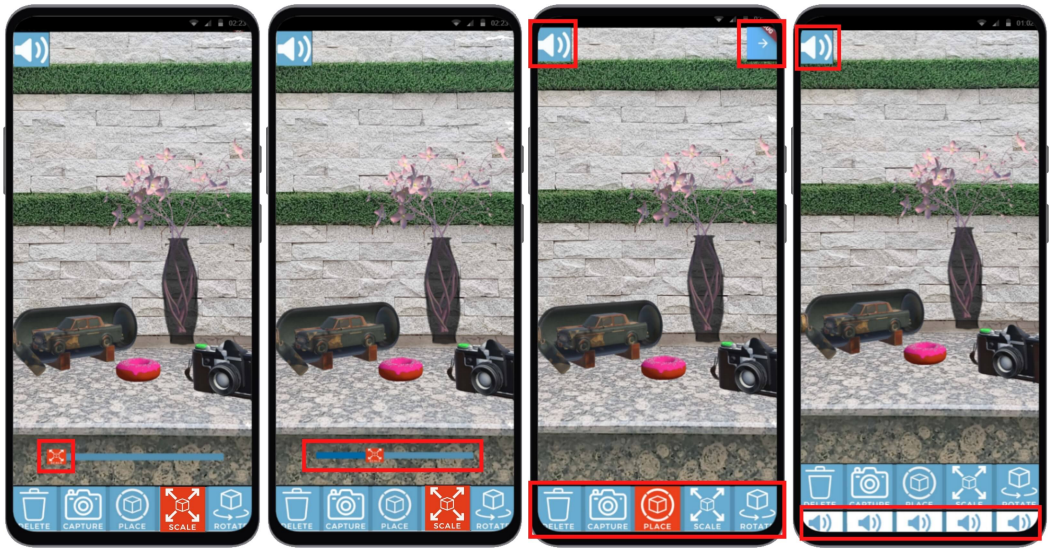
(i) Buttons filled across the screen for easy interaction with icons and text to explain each action

(ii) Semi-bold sans serif font used to increase text readability

(iii) Maximum space to view the generated painting when applying style transfer

(iv) Large style filter images for easy selection of painting style

Figure 3.4: AST Interface - Iteration One



(i) Increased knob size for easy interaction while dragging the slider

(ii) Color coded slider to provide feedback with interaction

(iii) Consistent button styles for easier identification

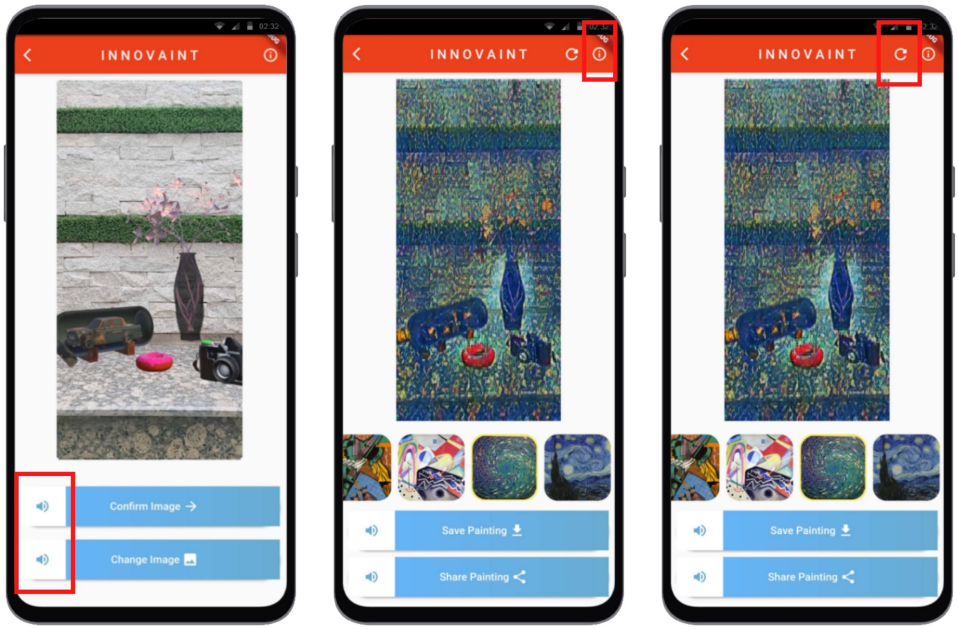
(iv) Speaker buttons to read-aloud text on buttons

Figure 3.5: AR Interface - Iteration Two

Artistic Style Transfer Interface

Speaker buttons were added to all the action buttons to support text-to-speech. Moreover, information buttons were clearly included in the app bars for easy access to the help section.

Lastly, an option to revert to the original image was also added. These additions can be seen in figure 3.6.



(i) Speaker button to read-aloud text on the buttons

(ii) Access to help section for demonstration of the application

(iii) Choice to revert back to the original image after applying style transfer

Figure 3.6: AST Interface - Iteration Two

3.5 Artistic Style Transfer Neural Network

The deep neural network used for applying the style transfer to the captured scenes is an open-source model from Tensorflow Hub [43], obtained under the Apache 2.0 License [63]. The model is part of the 'Magenta' research project [64] and is titled 'arbitrary-image-stylization-v1-256'. The TensorFlow Lite variant has been integrated into the application. The magenta style transfer model allows the style of any arbitrary image to be applied to a content image provided by the user by blending the content image in the style of the style image or 'transferring' the style to the content image.

The artistic style transfer model consists of the following two sub-models

Style Prediction Model

The style prediction model is applied to the style image to extract the style features from it. Hence, making it flexible enough to use any image as a style image. The style image size is 256

by 256 pixels with three colour channels in each pixel, resulting in a shape of 256, 256, 3. The colour space for the channels is RGB (red, green, blue). This model outputs a style bottleneck vector of the shape 1, 1, 100. A bottleneck vector represents the consolidated features that form the style of the image and could be thought of as a fingerprint for an image. The style prediction model is based on the MobilenetV2 [65] neural network. The MobilenetV2 model is based on a Convolutional Neural Network architecture and is optimized to enhance performance on mobile devices.

Style Transform Model

The style transform model is responsible for applying the style to the user image, constituting the actual 'style transfer' part. The model takes two inputs: 1) the user image to be stylized 2) the style bottleneck vector calculated from the style image. The user image is 384 by 384 pixels and consists of the same three colour channels (red, green, blue) for each pixel. Hence, the shape of the user image comes to be 384,384, 3. The output is a generated image that is styled to match the style image input. The size of this styled image is also 384 by 384 pixels with three RGB colour channels in a single pixel, making the shape the same as the user image – 384,384,3.

The model architecture is summarized in figure 3.7

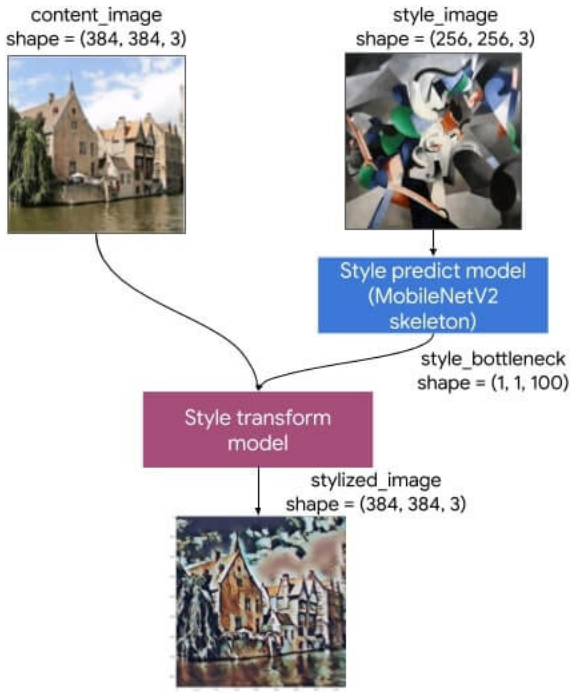


Figure 3.7: AST Architecture [66]

3.5.1 Model Evaluation

Before integrating it into the application, the original model implemented in TensorFlow available at [64] was tested to evaluate the usability. To run the model, two images were needed – a content image and a style image. For the content images, stock images that contained everyday objects and scenes were obtained. The rationale for choosing these content images was to mimic the augmented scenes that users would create while using the application as closely as possible. For the style images, the same painting style filters that were to be included in the application were used.

Figure 3.8 shows the results of applying the magenta artistic style transfer model to different combinations of images.

Figure 3.8: AST samples using the Magenta model

3.6 Functional Requirements

This section details the application's functional requirements categorized according to the interfaces—an asterisk (*) is placed at the end of requirements included in the second design iteration.

3.6.1 Augmented Reality Interface

- The application shall allow the user to browse through objects

- The application shall allow the user to select an object from the scroll view
- The application shall allow the user to detect a plane for placing objects
- The application shall allow the user to place an object in the scene
- The application shall allow the user to place multiple objects in the scene
- The application shall allow the user to delete a placed object
- The application shall allow the user to select a placed object
- The application shall allow the user to scale an object
- The application shall allow the user to rotate an object
- The application shall allow the user to drag an object across the plane
- The application shall allow the user to capture the created augmented scene
- The application shall allow the user to toggle user interface elements

3.6.2 Artistic Style Transfer Interface

- The application shall allow the user to select a content image for style transfer
- The application shall allow the user to select the last captured image
- The application shall allow the user to browse through painting styles
- The application shall allow the user to apply a painting style to the image
- The application shall allow the user to view the generated painting
- The application shall allow the user to save the generated painting
- The application shall allow the user to share the generated painting
- The application shall allow the user to revert to the original image (*)
- The application shall allow the user to receive auditory feedback on saving the image (*)

3.6.3 General

- The application shall allow the user to access the help section (*)
- The application shall allow the user to interact with the demo videos (*)
- The application shall allow the user to listen to any text displayed on-screen (*)

3.7 Non-Functional Requirements

This section details the non-functional requirements for the application.

3.7.1 Look and Feel Requirements

The Interface

- The application shall have a minimal interface.
- The application shall use a font that is easily readable by the user.
- The application shall appear authoritative.
- The application shall not use extra bright colour schemes.
- The application shall have a well-organized interface.

The Style of the Application

- The application shall have a clear call to actions.
- The application shall clearly show the navigation at the top.

3.7.2 Usability Requirements

Ease of Use

- The application shall validate all user entry
- The application shall allow easy browsing and selection of objects
- The application shall allow easy browsing and selection of styles
- The application shall be easy to use for people with only primary education

Ease of Learning

- The application shall provide a detailed tutorial section
- The application shall provide a help guide link with every action
- The application shall clearly state the purpose of its use

3.7.3 Performance Requirements

Speed Requirements

- Any interface between a user and the automated application shall have a maximum response time of 2 seconds
- The application shall have minimum animations for quicker responses

Reliability and Availability Requirements

- The application shall be available for use throughout the year.

3.7.4 Operational Requirements

- The application shall run on the Android OS
- The application must have access to the camera
- The application must have access to file storage
- The application shall work with versions of android above 9.

Chapter 4

Implementation and Methodology

4.1 Methodology

The project was initiated by doing a detailed analysis of the literature and related work, thoroughly studying the best techniques and technologies for developing the application, the interactions of people with aphasia with technology and communicative therapy using art. This was followed by a design phase consisting of the application front and back end designs. The implementation was initiated by developing the augmented reality interface first, then testing out the artistic style transfer model and finally creating an interface for the artistic style transfer and the entire application in general. Integration of the augmented reality interface and the artistic style transfer process was a vital step that followed, and the initial application prototype was built. This prototype was then put forward for proxy evaluations. The feedback by each proxy was incorporated into the application to make it more user friendly and accessible for people with aphasia. As a result, producing the final working prototype, which was finally tested. Figure 4.1 abstractly shows the stages of the project.

The agile methodology was applied as the software process model or methodology throughout the project. This was primarily chosen due to its flexibility with making changes to the initial requirements, as the project depended on feedback from experts to be incorporated into it. Hence, it seemed to be a wise decision to adopt the agile methodology.

Moreover, agile combines both incremental and iterative development processes. So, a detailed work breakdown structure was developed where tasks were broken down into smaller

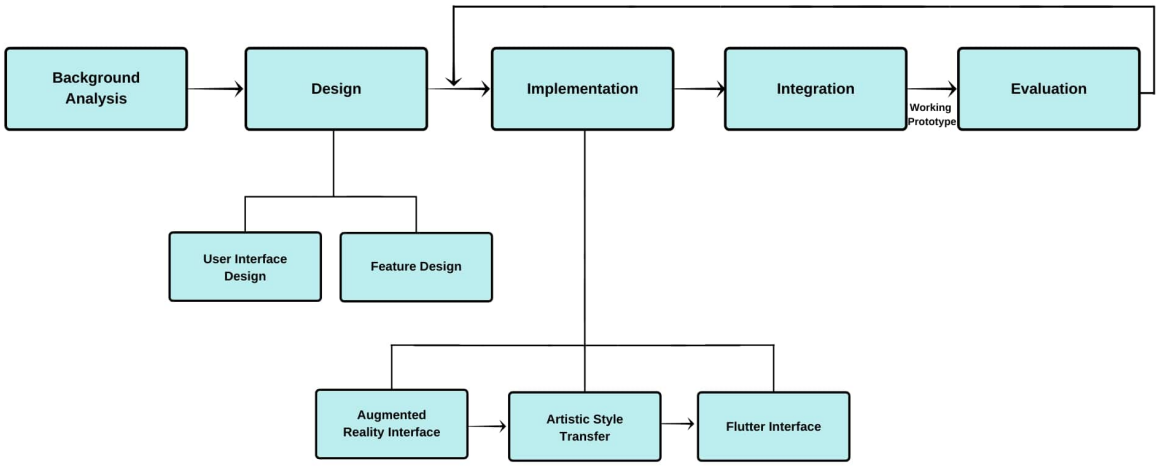


Figure 4.1: Project Methodology Flowchart

modules, and each module was worked on separately before integrating into the larger system. The traditional phases of planning, analyzing, designing, development and testing were applied to each module separately to produce the maximum quality of work.

4.2 Development Environment

This section details the environments used to develop the application and the rationale for choosing them.

Unity

Unity3D version 2019.4.3 was used to create the Augmented Reality interface. Unity was specifically used due to its flexibility with creating AR-based applications and was a natural choice due to its wide popularity. Moreover, the plugin to integrate the AR interface into flutter was available for the Unity Engine. The specific version 2019.4.3 had to be chosen for development due to the plugins integrated into the application, which did not support higher versions.

Unity ARFoundation

Unity ARFoundation framework is essentially a layer of abstraction for the ARCore SDK and supports features available in ARCore. After enabling support for ARFoundation in the application, features of ARCore could be accessed by adding ARCore as a package in Unity. Hence, ARFoundation has been used to develop the application due to ease of use. Version 2.1.18 of ARFoundation has been used due to compatibility with Unity 2019.4.3. Figure 4.2 below shows the architecture of ARFoundation.

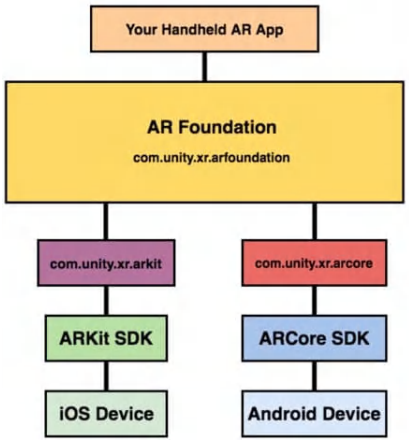


Figure 4.2: ARFoundation Architecture [67]

TensorFlow Lite

The artistic style transfer model used was developed in Python and is an open-source model by TensorFlow. The TensorFlow Lite framework has been chosen as it is optimized for mobile devices and would be the ideal choice to deploy deep learning models for on-device inferences. The TensorFlow Lite framework is considerably faster than TensorFlow and works well with devices with limited GPUs. [1] Lastly, it supported the android platform and was easily integrable with the flutter framework using packages.

Flutter

The Flutter framework was used to develop the overall application and as a front end to the artistic style transfer model because of its clean support for integration with Unity. Hence, enabling a seamless experience for the user throughout the entire application. The latest flutter version – 2.2, was used to develop the application to leverage all the latest features and support.

Android Operating System

The Android Operating System (OS) was chosen for the prototype development and evaluation, as most mobile users are on the Android OS at approximately 70.68% of the smartphone users [11]. This application is compatible with all Android versions greater than Android 9 and Android Software Development Kit (SDK) over version 28.

4.2.1 Technical Specifications

This section details the specific technologies used for the development of the components.

Augmented Reality Interface

Table 4.1 summarizes the technical details of the augmented reality interface.

Engine	Unity 3D	Version – 2019.4.3
Languages	C# .Net	
Framework	ARFoundation	To enable AR features support
SDK	ARCore	
Package	ARSubsystems	
Plugins	LeanTouch	For object interactions
	Flutter Unity View Widget	For exporting project to flutter
	Unity Native Gallery	For saving screenshot to gallery

Table 4.1: AR Interface Technical Specifications

Artistic Style Transfer Neural Network

Table 4.2 summarizes the technical details of the augmented reality interface.

Flutter Interface

Table 4.3 summarizes the technical details of the augmented reality interface.

Appendix A details the sources of the plugins and packages used.

Platform		TensorFlow Lite	
Language		Python	
Model	**Library**	Magenta	Open-source library for music and image generation and manipulation
	Name	arbitrary-image-stylization-v1-256/prediction	Model to use any arbitrary style image
		arbitrary-image-stylization-v1-256/transfer	Model to apply the style transfer
	Source	TensorFlow Hub	

Table 4.2: AST Model Technical Details

IDE	Android Studio	Version – 3.5.3
Framework	Flutter	Version – 2.2
Languages	Dart	For Flutter
	Kotlin	For Android
Packages	Flutter Unity Widget	To embed augmented reality interface
	Flutter Bloc	For UI state management
	Tflite Flutter	To connect with TensorFlow Lite and enable on-device inference
	Freezed Annotation	To generate immutable classes
	Multi Image Provider	To pick images from the device
	Path Provider	To manage file paths on the application
	Image Gallery Saver	To save paintings to the device
	Share Plus	To share paintings externally
	Flutter TTS	To enable read-aloud functionality
	YouTube Player	To embed demo videos from YouTube

Table 4.3: Flutter Interface Technical Details

4.3 Implementation Details

4.3.1 Project Modules

Figure 4.3 summarizes the main and the submodules developed for the application.

Augmented Reality Interface

This module consists of designing and developing the interface to allow object interaction in the augmented environment.

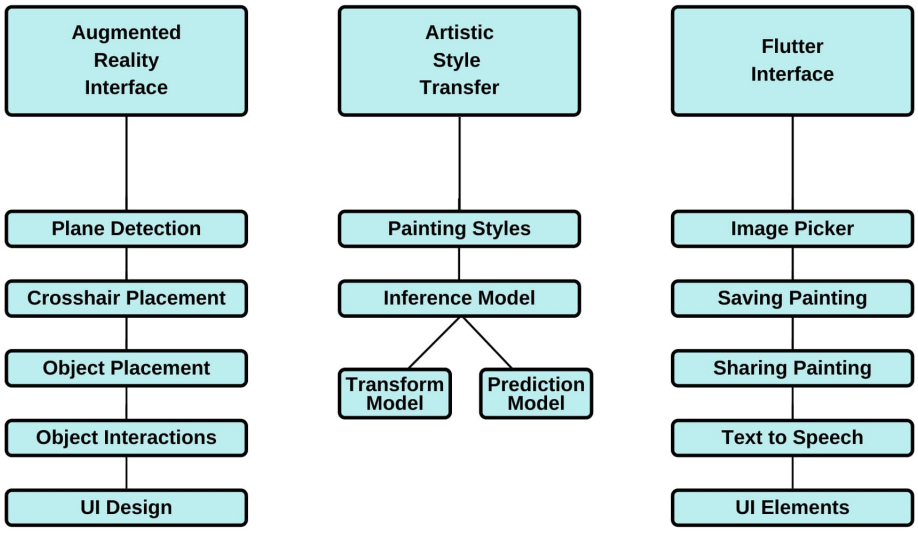


Figure 4.3: Project Modules and Sub-Modules

Plane Detection: Detect a horizontal AR plane in the real environment to place objects upon

Crosshair Placement: Marker integration to place objects at precise locations

Object Placement: Instantiating new virtual objects in the environment

Object Interactions: Selecting, Deleting, Rotating, Scaling objects in the environment

UI Design: Design of user interface elements to enhance accessibility

Artistic Style Transfer

This module includes the style transfer model and designing the related painting styles.

Painting Styles: Designing the style filters to transform pictures into artistic styles

Inference Model: Style Transfer Neural Network which combines the content of one image with the style of another using two sub-models – Prediction Model and Transform Model

Flutter Interface

This module includes the design and development of a front-end for the style transfer and to connect the AR interface with it.

Image Picker: To allow the user to change images and select one from the gallery

Saving Painting: Save the generated painting to the gallery

Sharing Painting: Share the generated painting to third-party applications

Text-to-Speech: Read aloud any displayed text

UI Elements: Designing an accessible user interface

4.3.2 Augmented Reality Interface

The AR interface is the main application interface where the user interacts with virtual objects in the real space to create an augmented environment. The application was built in Unity using the ARFoundation and ARCore development kits to enable Augmented Reality support. The 'flutter unity view widget' plugin embeds the unity interface into the flutter application.

This interface enables users to select from a range of virtual objects and place them in the scene, using a crosshair for accurate positioning of the object and providing a reference for the user for the placement. The code snippet in listing 4.1 is used to instantiate objects. The update function is run once in every frame of the application. It checks the position of the touch on the screen and places the crosshair at that exact position, ignoring all the touches over the interface elements. There are three phases of touch that are being checked for:

Start: The touch phase is started when the user first touches the screen. An object is selected if a raycast is hitting an object with the tag 'Spawnable' and instantiated if there is no object at that position. To instantiate the object, the prefab of the selected object from the object's scroll view is loaded and placed in the scene.

Moved: The touch phase is moved when the user slides or moves the finger across the screen. In this phase, the selected object is dragged along the screen by updating the position of the object to match the touch position.

Ended: The touch phase ends when the user lifts the finger from the screen, and the selected object is made null to avoid any unintended interactions on the objects.

```
1  // Update is called once per frame
2  void Update()
3  {
4    // Recalculate Crosshair Position
5    CrosshairCalculation();
6
7    touch = Input.GetTouch(0);
8
9    // Ignore Touch over UI Element
10   if (IsPointerOverUI(touch)) return;
11
12   spawnablePrefab = DataManager.Instance.GetModel();
13
14   if (Input.touchCount < 0)
```

```
    return;

  RaycastHit hit;
  Ray ray = arCam.ScreenPointToRay(Input.GetTouch(0).position);

  // If a placed object is touched - Select Object
  // If there is no placed object - Place New Object
  // If finger is sliding - Drag Object

  if (_raycastManager.Raycast(Input.GetTouch(0).position, _hits))
  {
    if (Input.GetTouch(0).phase == TouchPhase.Began && spawnedObject == null)
    {
      if (Physics.Raycast(ray, out hit))
      {
        if (hit.collider.gameObject.tag == "Spawnable")
        {
          spawnedObject = hit.collider.gameObject;
        }
        else
        {
          spawnedObject = Instantiate(DataManager.Instance.GetModel(), pose.
    position,
            DataManager.Instance.GetModel().transform.rotation);
        }
      }
    }
    else if (Input.GetTouch(0).phase == TouchPhase.Moved && spawnedObject !=
    null)
    {
      spawnedObject.transform.position = _hits[0].pose.position;
    }

    if (Input.GetTouch(0).phase == TouchPhase.Ended)
    {
      spawnedObject = null;
    }
  }
}
```

Listing 4.1: Object Placement Function (adapted from [1] [2])

The user can interact with the objects in the following ways:

Dragging

The object could be dragged or moved around in the scene to position it better or change the position once by selecting it once it has been placed. The drag interaction requires one finger to be able to move the object around.

Scaling

The object could be scaled to make it bigger or smaller in size after it has been placed. It requires. Scaling could be done in two ways: 1) by using gestures that require a multi-finger touch (two fingers) by pinching to scale up or scale down and 2) selecting the object and using a slider to scale the object as required, which requires a single finger interaction.

Rotating

The object could be rotated along the x-axis within a range of 0 to 360 degrees, covering the entire turn. Rotation can also be done in two ways: 1) by using gestures that require a multi-finger touch (two fingers) by moving the fingers circular motion in a clockwise or anticlockwise direction and 2) selecting the object and using a slider to rotate the object in a complete turn which requires a single finger interaction.

Scaling and rotation using gestures are handled by the Lean Touch plugin, which provides components to add these features to the objects. The code in listing 4.2 shows the scaling and rotation scripts for using sliders. A singleton pattern has been used in the sliders class to allow the selected object reference to be passed to this class when an object is clicked. This ensures that only one object is being interacted with at a single time. The scale interaction has a minimum value of 0.1 and a maximum value of 3 to keep the objects between a reasonable size. A listener is set to increase or decrease the object's size whenever there is a change in the scale slider value. The rotation scale interaction has values between 0 and 360 degrees for a complete axis turn. Another listener is set to change the rotate the object whenever there is a change in the rotation slider value.

```
1 // Define Singleton
2 private void Awake()
3 {
4   if (Instance == null)
5     Instance = this;
6   else if (Instance != this)
7     Destroy(gameObject);
```

```
}

// Set Min Max Values and Slider Listener
void Start()
{
  scaleSlider.minValue = scaleMinValue;
  scaleSlider.maxValue = scaleMaxValue;
  scaleSlider.onValueChanged.AddListener(ScaleSliderUpdate);

  rotateSlider.minValue = rotMinValue;
  rotateSlider.maxValue = rotMaxValue;
  rotateSlider.onValueChanged.AddListener(RotateSliderUpdate);
}

// Update Object Size
void ScaleSliderUpdate(float value)
{
  Object.transform.localScale = new Vector3(value, value, value);
}

// Update Object Angle
void RotateSliderUpdate(float value)
{
  Object.transform.localEulerAngles = new Vector3(transform.rotation.x, value,
    transform.rotation.z);
}
```

Listing 4.2: Transformation Sliders Functions (adapted from [3])

Deleting

The object could be deleted after it has been placed in the scene by selecting it and pressing the delete button. This interaction requires a single finger touch. When the delete button is pressed, it checks the selected object and deletes it using the command: *destroy*(*gameObject*)

Finally, the interface allows users to capture the augmented environment scene once it is finalized or take multiple snapshots while they progress. The CaptureScreen() function in listing 4.3 is used to take the snapshot. When a scene is captured, all the user interface elements and the planes are momentarily (for 0.3 seconds) disabled so that the scene is aesthetically pleasing without any extra elements in the image. When the capture button is pressed, the Screen Capture component of Unity is used to save the captured image to the gallery. Also, a trigger

is sent to the flutter interface, so it activates the 'Paint' button for applying the style filters, using a component from the Flutter Unity View Widget plugin. Lastly, the Native Gallery plugin has been used to save the image to the gallery.

```
// Disable UI Elements, take Snapshot and Save to Gallery, Inform Flutter
private void CaptureScreen()
{
  GetComponent<UnityMessageManager>().SendMessageToFlutter("capture");

  arPlaneManager.enabled = false;
  foreach (ARPlane plane in arPlaneManager.trackables)
  {
    plane.gameObject.SetActive(false);
  }

  ScreenCapture.CaptureScreenshot("arScene.png");

  canvas.gameObject.SetActive(false);
  marker.gameObject.SetActive(false);

  Invoke("EnablePlane", 0.3f);

  string existingPath = Application.persistentDataPath + "/arScene.png";

  NativeGallery.Permission permission = NativeGallery.SaveImageToGallery(
    existingPath, "unityApp", "arSc.png",
    (success, path) => dt2.text = "Media save result: " + success + " " + path);
}

// Enable UI Elements
void EnablePlane()
{
  arPlaneManager.enabled = true;
  foreach (ARPlane plane in arPlaneManager.trackables)
  {
    plane.gameObject.SetActive(true);
  }

  canvas.gameObject.SetActive(true);
  marker.gameObject.SetActive(true);
}
```

Listing 4.3: Function to Capture Scene

Virtual 3D Objects

All the virtual objects have been acquired from 3d model libraries online under the Creative Commons license. The main sources are:

- Sketchfab (https://sketchfab.com)
- Free 3D (https://free3d.com)

Appendix B details the set of objects included.

The models have been imported into Unity. Then the layers of the models have been built – starting with the shape, applying the texture, and the material to be used for the texture to render the object finally. An original prefab of each object has been created to allow it to be instantiated. The following components have been added to each of the prefabs:

- Box Collider – Check if a Raycast is hitting the object
- Lean Selectable – Select the object (LeanTouch)
- Lean Drag Translate – Drag the object (LeanTouch)
- Lean Pinch Zoom – Scale the object using gestures (LeanTouch)
- Lean Rotate Axis – Rotate the object using gestures along an axis (LeanTouch)
- Slider Script – Scale and Rotate the object using sliders

4.3.3 Artistic Style Transfer

A typical flow of applying the style transfer end to end to a new image begins by preprocessing both the content image and the style image to satisfy the shape conditions. This is followed by calculating a style bottleneck from the style image using the prediction model and finally applying the transform model to apply the style bottleneck to the content image.

Figure 4.4 shows a flowchart of the technical process for applying the artistic style transfer.

The tflite package has been used to integrate this model into flutter, enabling support for using TensorFlow models and APIs in flutter. The lite versions of both the sub-models – 'transform model' and 'prediction model' have been downloaded from the TensorFlow hub [43] and added in the application as flutter assets. This enables on-device inferences even when an internet connection is not available, enhancing performance.

The transfer function used in the application takes as input the content image and the style image as Uint8List type and returns a stylized image as a Uint8List object which is converted to

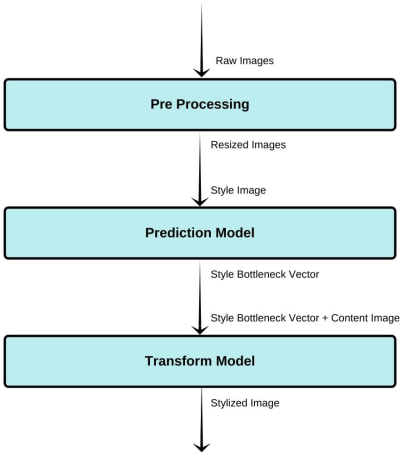


Figure 4.4: AST Algorithm Flowchart

an image to be viewed and saved. Next, the function reshapes both the images to match the size requirement of the model (256 256 3 for the style image and 384 384 3 for the content image). This is followed by calculating a style bottleneck from the style image using the prediction model. Finally, the transform model combines the style bottleneck and the content image to output a stylized image after altering it (rotation and flipping).

Painting Style Filters

The painting style images have been acquired from virtual galleries and stock libraries under the creative commons license. There are 20 different filters included. Appendix C lists all the painting styles in the application. Each image has been resized to 512 by 512 pixels and added through the assets library in the flutter application. They are all displayed in a list view for easy access to the user.

4.3.4 Flutter Application

The flutter application provides the user with an interface to access the Artistic Style Transfer model. This interface follows once the user is finished working in the augmented scene. The latest image captured from the AR interface is selected by default. The user is also given an option to change the image if any other image from the entire process needs to be chosen. The multi-image provider package has been included to allow the user easy access for accessing image files on the device.

After the image is finalized, the user is diverted to the style transfer screen, where the user can view the 20 painting styles and choose any of them to be applied to the selected image.

When a user chooses a painting style, both the style transfer models are called, and the image is stylized by combining the captured scene image with the selected style filter.

Once the user is satisfied with the output, they could either save the painting to the gallery or share the artwork with an external source on their smartphone. First, the file bytes are converted to an image and then saved to the gallery using the 'image gallery saver' package or shared using the 'share plus' package.

The code snippet in listing 4.4 displays the save and share functions.

```
// Save Function -  convert painting bytes and save to gallery
Future<String> saveImage(Uint8List bytes) async {
    final result = await ImageGallerySaver.saveImage(bytes);
    return result['filePath'];
}

// Share Function - convert painting bytes to image, save and share
Future saveAndShare(Uint8List bytes) async {
    final directory = await getApplicationDocumentsDirectory();
    final image = File('${directory.path}/styledImage.png');
    image.writeAsBytes(bytes);
    await Share.shareFiles([image.path]);
}
```

Listing 4.4: Save and Share Painting

Text to Speech

The 'flutter_tts' package has been used to integrate text-to-speech functionality into the application. The package provides a prebuilt speech synthesizer that supports multiple languages. The text-to-speech has been tied to every call to action and display text in the application. The code snippet in listing 4.5 here shows the implementation when the speaker icon is clicked to read the text on the screen. The language has been set to English. The speaking speed of 0.6 and the maximum volume ensures speech is as clear as possible for the user.

```
import 'package:flutter_tts/flutter_tts.dart';

final FlutterTts tts = FlutterTts();

tts.setLanguage('en');
tts.setSpeechRate(0.6);
```

```
7 tts.setVolume(1.0);

9 tts.speak('Innovaint. An AR Painting Application');
```

Listing 4.5: Text-to-Speech

A message manager has been used to communicate between Unity and flutter to integrate the text-to-speech in the augmented reality interface. A button click on the AR interface calls the TTS function from the flutter interface to speak the text.

Chapter 5

Proxy Evaluations

Two experts in human-computer interaction and speech and language therapy who had vast experience working with and designing technology for people with aphasia were involved in this study as proxies. They were included in the design process to help improve the application also asked to perform a final evaluation of the prototype. Proxies have been shown to provide valuable input in the past for designing for people with aphasia when actual user involvement is restricted [68].

As part of a formative evaluation, they were asked to provide feedback on the initial prototype of the application to finesse the application to enhance accessibility for people with aphasia. The interaction involved providing them with the application and requiring them to install it on their phones for testing it out. This was followed by a discussion to know their thoughts on the user interface and features design through a semi-structured interview to understand individual aspects better.

Finally, one of the proxies was asked to perform a final evaluation of the application after the changes from the formative evaluations were incorporated into the final prototype. This interaction also involved providing the proxy with the application to be tested on a mobile device. The evaluation involved discussing the improvements made to the application, possible extensions and using it as a tool for expression.

5.1 Formative Evaluation

Due to the explorative nature of this study, a thematic analysis of the interviews was conducted, where the data was divided according to the topics under discussion. Thematic analysis is a

flexible method to divide extensive, unorganized data in a well-structured manner [69].

The main themes drawn from the interviews were: usability, therapy, objects and styles, accessible design, user interface, features, and extensions.

The interview transcripts can be found in section D.1 of Appendix D.

5.1.1 Usability by People with Aphasia

The proxies were confident that the application is usable for people with mild to moderate aphasia.

> "with my experience of watching people interact with different technologies, I think some people will be able to pick up, and sort of know what to expect with it, and others will get the hang of it with a bit of trial and error, which is often the way PWA need to approach technology anyway."

Although they were optimistic that the application is usable for people with severe aphasia as well based on previous studies however could not precisely be sure without user studies as reported:

> "I am less confident about being a judge for PWA but won't rule them out because I have seen someone with very severe aphasia effectively using technology in the past."

Another factor that highly influenced the usability was the measure of tech literacy and comfortability which varies per individual, as pointed out by them:

> "thing about people with aphasia is that some of them are far more comfortable with technology than others."

As a result, some people find it easier to grasp the usage quicker. On the other hand, others need some time and attempts to use the application before smoothly utilizing it.

Other factors that impact the interaction are whether a person has any further disabilities because of the stroke or injury.

> "This difference around severe, mild and presence of other disabilities on top of that like whether there is a physical ailment or cognitive ailment, some people might find it very hard to remember the process of the application."

5.1.2 Usability as Expression in Therapy

The application could practically aid the type of speech therapy that falls under ”total communication therapy or compensatory communication,” which trains people to find alternative ways of sending their message across.

> ”This represents augmentative and alternative communication which is a way of people using skills that they have to covey something to express themselves in a different way and so what you might explore with someone in a therapeutic context with what you have created is what side of you can you express with this”

Additionally, it could be used as a tool for expression.

> ”How you could use this to create something that might spark a conversation that you could use as a conversation point with others that you could show to others to express a feeling or a thought that would allow you to start a conversation with somebody, what this would allow them to do in a therapeutic sense is to learn how to augment an existing thing to give it more communicative or expressive value and tailor it in a way that it becomes a way of expressing their identity, a way of supporting somebody express and develop and maybe express a creative identity that they might not be able to otherwise convey and they might be able to hone that and refine that over time and find their own voice.”

Apart from the therapeutic use of the application, it could equally be used for curating aesthetic paintings.

> ”it could be used to kind of play or just being creative.”

5.1.3 Design for People with Aphasia

They also pointed out how design considerations for people with aphasia had been catered to in the application design.

> ”I have used it one-handed, and I can see that is obviously a consideration you have put in while using it, which is nice.”

> ”Once they (PWA) get to know it, the interactions you have put in place are quite observant to good design principles for someone with aphasia, you know the consistency, minimizing the cognitive load and things like that.”

Moreover, people with aphasia could use the application with different levels of complexity, and even with minimal input, a user could successfully curate scenes and generate paintings.

> ”as you become more accustomed to it and want to explore it more, there is a variety of ways you could start to tailor and hone and assemble and curate your image, and it’s really nice that it’s got an opportunity for success at different levels of complexity.”

They also mentioned how a choice of the device would be convenient for the users.

> ”everybody with aphasia is different; some people will benefit from using it on a tablet, some people will benefit from using it on a phone their access will be enhanced on using it on a phone than a tablet.”

5.1.4 Objects and Styles

The proxies also expressed positive remarks on the selection of objects and style filters included in the application.

> ”I enjoyed seeing the playful variety of objects; the variety makes it sort of an appealing thing. It is also interesting to have a variety of shapes and sizes of objects.”

and

> ”The selection of filters is beautiful; they speak for themselves; they don’t need a word attached to them. They look very attractive.”

5.1.5 User Interface Design

Feedback on the design of the augmented reality interface was also favourable.

> ”what’s really nice about the slider is when someone is focused, it zooms in and that each icon has a text explainer with it and highlights when it is selected, making it understandable.”

> “quite nice size buttons which someone could interact with”

> ”it’s a really straightforward interface I can play with.”

Similarly, the evaluators also appreciated the style transfer interface

"I like the fact the confirm image has an arrow next to it. It makes me think that's the main thing I need to do next."

"The main thing is the interface is very uncluttered, which is a nice feature."

There were also suggestions on improving the interface. These included improving the slider and the objects' scroll view to reduce precise touch positions.

"The slide bar button is quite small to click and drag from, just increasing the size to require less precision."

"The scroll view has an indefinite setting point"

5.1.6 Features

The features identified during the discussions included adding a Text to Speech functionality as the therapist believed:

"The read-aloud function is something some people really benefit from; definitely adding in a Text-To-Speech is a support."

Moreover, a help section was suggested by the proxy to inform users of the functionality and features of the application:

"It might need some introduction for people using it for the first time, and a video demonstration will really help people support what they could do with it."

5.1.7 Extensions

They also suggested possible extensions to the application where a more extensive set of objects and style filters could be added to express more of themselves.

"objects could be used to support people to convey a sentiment to create something that is humorous or emotive or express frustration or anger, and similarly with the filters, some of them could be really black and white or dark red."

In addition to these, objects of reference are worth exploring

"in speech and language therapy where people have no language, you could use what's called objects of reference to represent whatever is that you want to talk about which can become part of the communication system if you like where the object represents a concept that you want to talk about."

They also suggested adding more flexibility to the style transfer where the user could layer filters on top of one another. There could be options to control the blending of the content and style images.

> "the image could be a bit more like the style or a bit less like that, I'd like to retain a bit more of the original image, or I'd like to obscure it a bit more."

Finally, the proxies were interested in making the application available on a broader scale to make more people benefit from it and wanted to know

> "whether it might be possible to post it on play store or anything like that."

They also hoped to see the application being evaluated with actual people with aphasia in user studies.

5.2 Improvements to the Application

Based on the feedback by the proxies, feature and design improvements were made to the application.

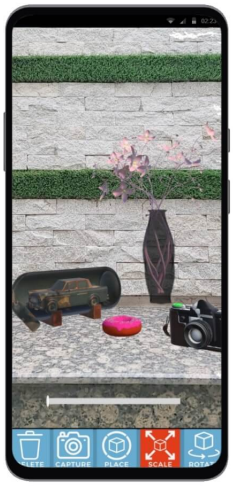
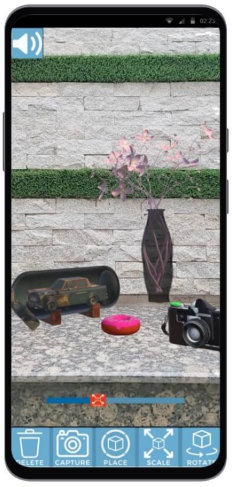
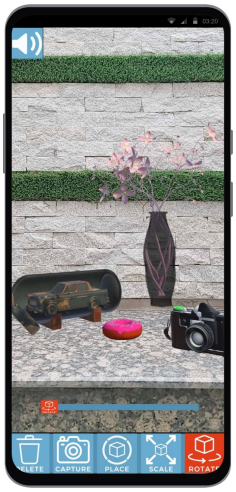
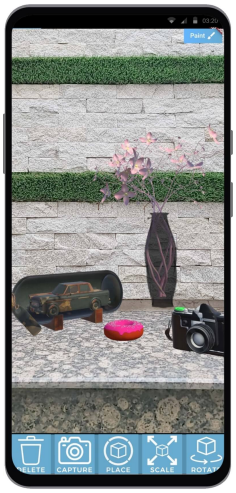
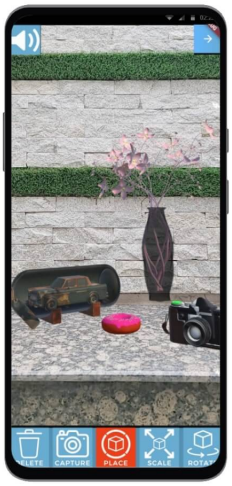
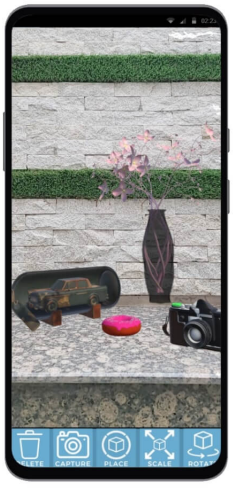
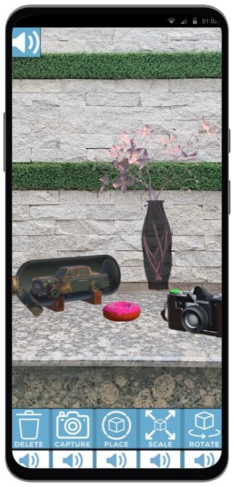
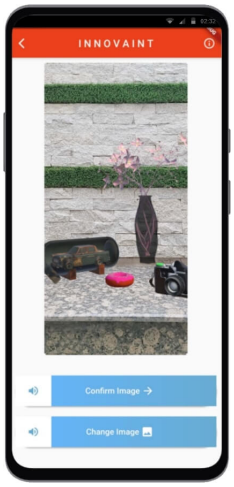
Text-to-Speech functionality for people who have trouble reading was added. So, for every call to action and text displayed in the application, the user should have an option to have it read out loud.

Moreover, to accommodate people who are not very comfortable using technology, a help section was included within the application, which contained detailed instructions and a demonstration of how to use the application.

Another feature added was viewing or going back to the original curated scene image while applying the different painting styles.

The user interface was also enhanced to make it more accessible. This included making changes to the interface so that all buttons on a single screen look the same to make identification consistent. The scale and the rotate sliders were connected to their respective call of actions to reemphasize their tasks. Moreover, the size of the knobs of these sliders was increased to make them easier to slide and interact with. Lastly, the sensitivity of the objects' scroll view was decreased to increase accessibility.

Figure 5.1 presents these improvements in comparison to the first design iteration.

Initial Prototype	Prototype with Improvements	
		
Plain Slider	Slider Accessibility Improved	
		
Mixed Button Styles	Consistent Button Styles	
		
No Text-to-Speech	Text-to-Speech functionality included	

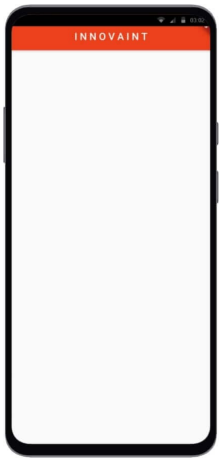
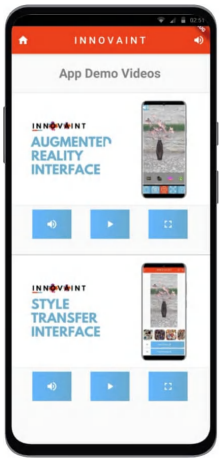
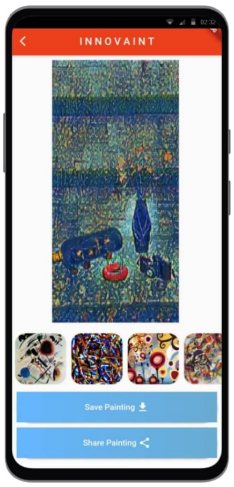
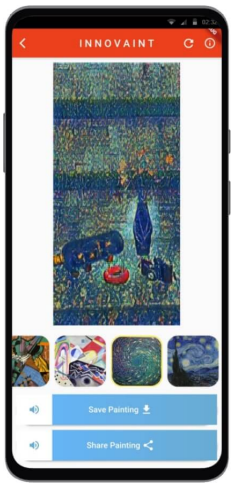
Initial Prototype	Prototype with Improvements
	
No Help Section	Help Section Included
	
Can't Revert to Original Image	Option to Revert to Original Image added

Table 5.1: Comparison of Prototype Versions across Design Iterations

5.3 Final Evaluation

The proxy believed the additions and improvements made to the application were productive, both in terms of the features added and the user interface design. The interview transcript can be found in section D.2 of Appendix D. The updated version of the application was a prominent improvement over the previous iteration, as the proxy reported:

> "I am really impressed with the amount of things you managed to implement"

The accessibility of the application was enhanced further by adding the Text-to-Speech functionality and the help section, as the proxy believed:

> "I think on a very practical level the TTS will help, will increase the accessibility

and will be very well received; that's real improvements on the access."

Likewise, the way the help section is exhibited would support the users.

"The help section will add support; a video demonstration is a constructive way of presenting it than a page of written text, so I like the fact that you have considered that presentation style in your help menu."

Moreover, the improvements to the slider design were notable.

"The fact that it ties those things (actions) together makes the interpretation of the slider clearer and more apparent to people now."

The objects added were relevant to aiding self-expression for the users

"In terms of promoting expression, saying something about yourself and providing opportunities for discussion around an image, I think the choice of objects you have introduced adds another opportunity for complex communication around the images, so that's a really nice feature."

Similarly, the increased list of painting style filters provided more freedom of expression without explicitly stating something.

"They could use that to convey something either implicitly or state I chose this one because its spiky and I feel angry about this, or they can just show it, and the person interpreting it could just experience something that feels more spiky without them necessarily having to articulate it."

Furthermore, the choice and presentation of these extended elements was significant

"I think it's really nice you have got that diversity of choices and the fact that it is repented in a less prescriptive and a more creative way. So, I think these are lovely additions."

The rationale behind the choice of objects and filters was another valuable aspect presenting more scenarios of expression and interpretation.

"The thought you put into the selection of objects and filters suddenly gives an extra dimension and makes it broaden how I could imagine people using it and the kind of the simulation it could provide"

Discussing the extensions to the application, the specialist believed that user studies with people with aphasia might illustrate possible feature additions.

> "It could well be sufficient but worth considering and exploring"

Actual user feedback on some components of the application would be beneficial. For instance, how access to the help section should be shown

> "It might be something you explore in user testing if that symbol is enough, or people will need additional text support to say demo; it's just a point of discussion and reflection."

The proxy also compared the application to existing applications used in communication therapy and believed this was a more encouraging way of expression for users.

> "There is not an obvious right or wrong answer to where you put your giant teddy bear chair, or there is no I am doing it wrong, or the picture I have made is wrong in the way there can be for medical devices where you are trying to get a right word to go in a slot to make you speak correctly. This is about pure freedom of expression."

Finally, the specialist believed the application to be a promising tool that has the potential for being used by people with aphasia for expression as an alternative way of communication.

> "This application offers a novel and stimulating tool to explore expression for people with aphasia, and it contrasts with medicalized apps because it gives the user more freedom to decide on their mode of expression."

5.4 Discussion

The specialists confirmed that the application would be usable for people with aphasia. This is in line with applications like Bangaten [70], MAT-APP [71], MakeWrite [13], and Inker [72] which investigated how people with aphasia interact with technology and obtained reassuring results. Further, Innovaint has the potential to be used as a tool for expression and aid communication therapy, using it as an augmentative and alternative communication device. Results from past studies such as the mHealth [12] and PhotoSearch [73] applications researching the use of computer-aided AAC devices have also been favourable and shown to aid linguistic skills for people with aphasia.

Innovaint takes a novel approach to allow self-expression for people with aphasia by providing an alternative way of communication. It is an attempt to aid social interaction creatively instead of using traditional functional applications that restrict or lead the user. Innovaint explores the connection between communication and art therapy, and the evaluations are promising. This project aimed to research this gap, which had been untapped until now and found potential for further exploration.

Furthermore, thoughts on equipping people with aphasia with Augmented Reality environments have also been optimistic. This project also provides new insight into using Artistic Style Transfer techniques constructively than solely for entertainment purposes. Innovaint is an original application that proposes Augmented Reality and Artistic Style Transfer technologies in the context of expression and as an accessible technology.

The results also established that the user interface had been carefully designed to support people with aphasia considering practices of designing accessible technology. Apart from the design, specialized features to aid users have been included, like the Text-to-Speech, providing multi-modal support. Moreover, the choice of elements available to curate scenes has been rationalized to promote expression. The range of virtual objects and medium of style filters included have been inspired from their use in therapy [6] [20], hence potentially offering similar benefits to a physical session. The use of technology instead of physical therapy sessions could open new dimensions.

Finally, it was beyond the scope of this project to perform user studies involving people with aphasia. Direct input and feedback from them would be a valuable addition to this study. Further research by including people with aphasia would help finesse the application by tailoring the existing functionality and adding more features.

Chapter 6

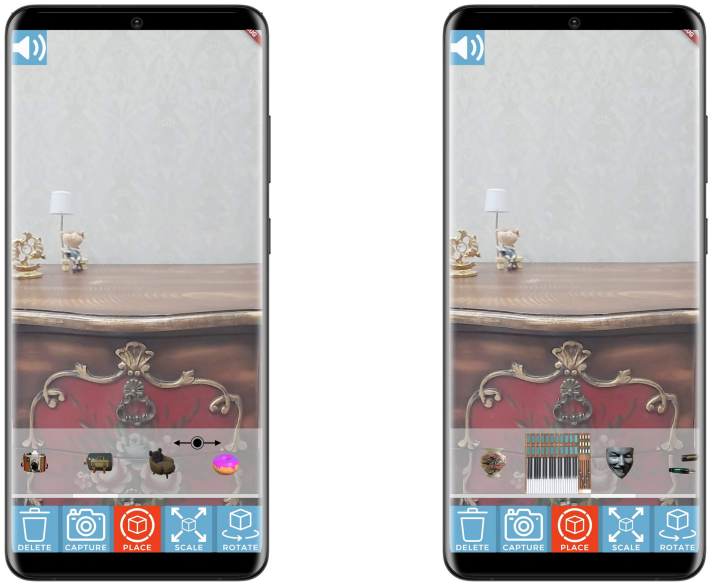
Innovaint: Application Overview

6.1 Primary Features

This section details the main application flow of a typical user session

6.1.1 Browse Objects

The user browses through the list of objects available by scrolling over the objects' scroll view (Figure 6.1).



(i) Browse through objects by sliding finger

(ii) All objects displayed in a scroll view

Figure 6.1: Feature: Browse Objects

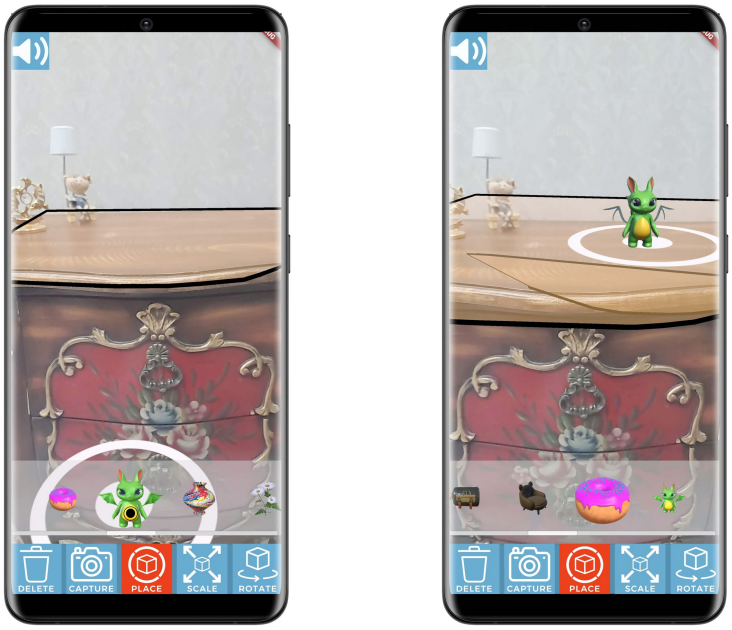
Figure 6.2: Feature: Plane Detection

6.1.2 Plane Detection

The user detects a horizontal plane to place the objects on by hovering the phone over any horizontal plane surface (Figure 6.2).

6.1.3 Place Object

The user selects an object (Figure 6.3i) and touches a point on the screen to instantiate it (Figure 6.3ii). The crosshair enables precise placement of the object.



(i) Select object image from scroll view

(ii) Object placed by tapping on screen

Figure 6.3: Feature: Place Object

Placing Multiple Objects

The user could place multiple objects in the augmented scene using the process defined above (Figure 6.4).

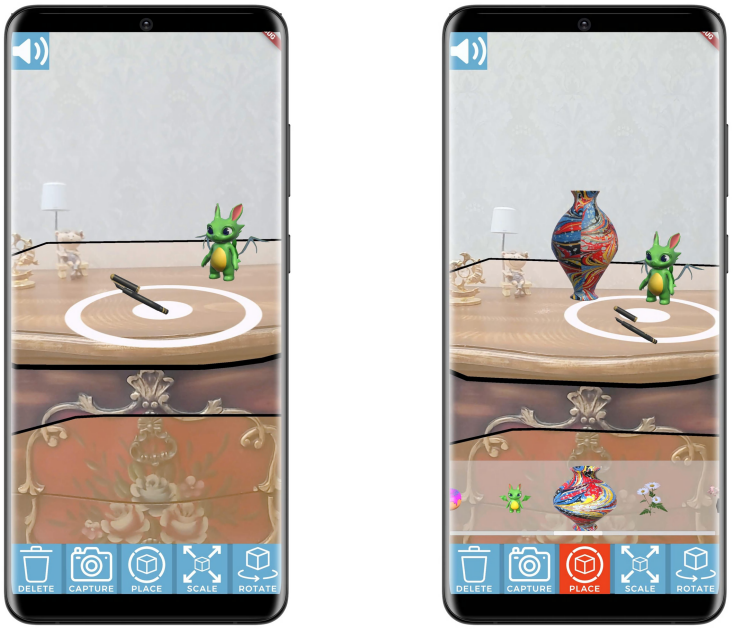


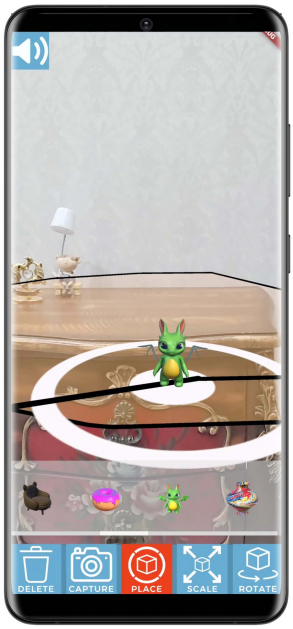
Figure 6.4: Feature: Place Multiple Objects

6.1.4 Drag Object

The user has two options to drag objects

Drag while Placing

The user starts dragging the object while placing it by sliding the finger over the screen while touching it (Figure 6.5).



(i) Drag object by sliding finger across screen

(ii) Remove finger from screen to place object

Figure 6.5: Feature: Drag while Placing

Select and Drag

The user selects an already placed object (Figure 6.6i) and slides the finger to drag it (Figure 6.6ii).

(i) Select object by tapping on it

(ii) Slide finger to drag across screen

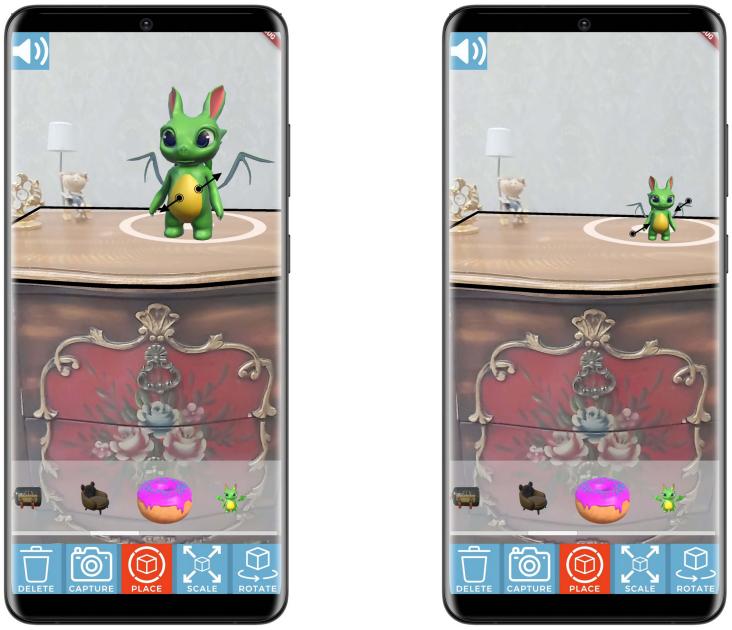
Figure 6.6: Feature: Select and Drag

6.1.5 Scale Object

The user could change the size of the objects through the following.

Scale using Gestures

The user uses the pinch gesture (two-finger interaction) to increase (Figure 6.7i) or decrease (Figure 6.7ii) the size of an object by changing the pinch size.



(i) Pinch to scale up

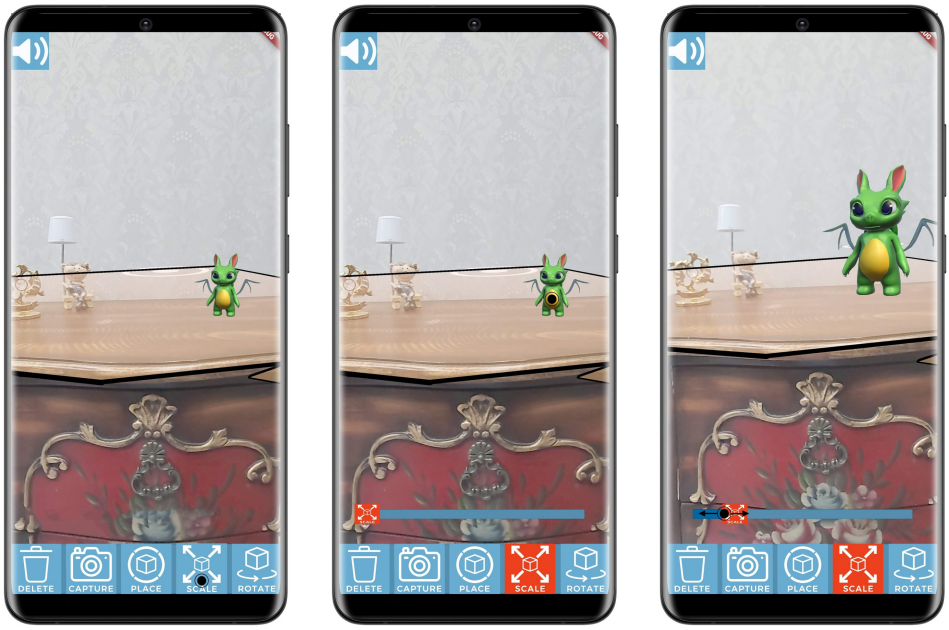
(ii) Pinch to scale down

Figure 6.7: Feature: Scale using Gestures

Scale using Slider

The following steps need to be performed to scale using a slider

1. The user presses the 'Scale' button to view the scale slider (Figure 6.8i)
2. The user selects an object by clicking on it (Figure 6.8ii)
3. The user slides through the slider to alter the size of the object (Figure 6.8iii)



(i) Press scale button to show slider

(ii) Select object by tapping on it

(iii) Slide through the slider to change size

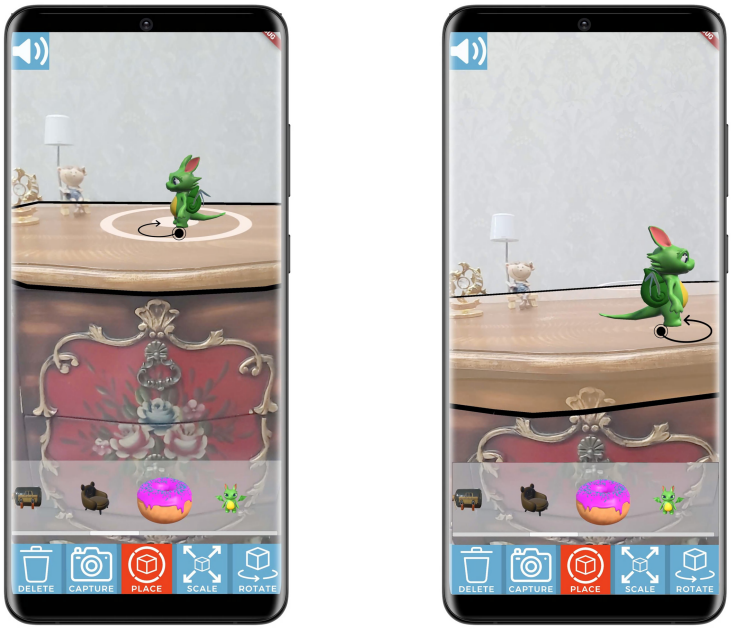
Figure 6.8: Feature: Scale using Slider

6.1.6 Rotate Object

The user could change the angle of the objects in the following ways.

Rotate using Gestures

The user uses a two-finger gesture in a circular motion to rotate the object along the x-axis (Figure 6.9)



(i) Object rotated left

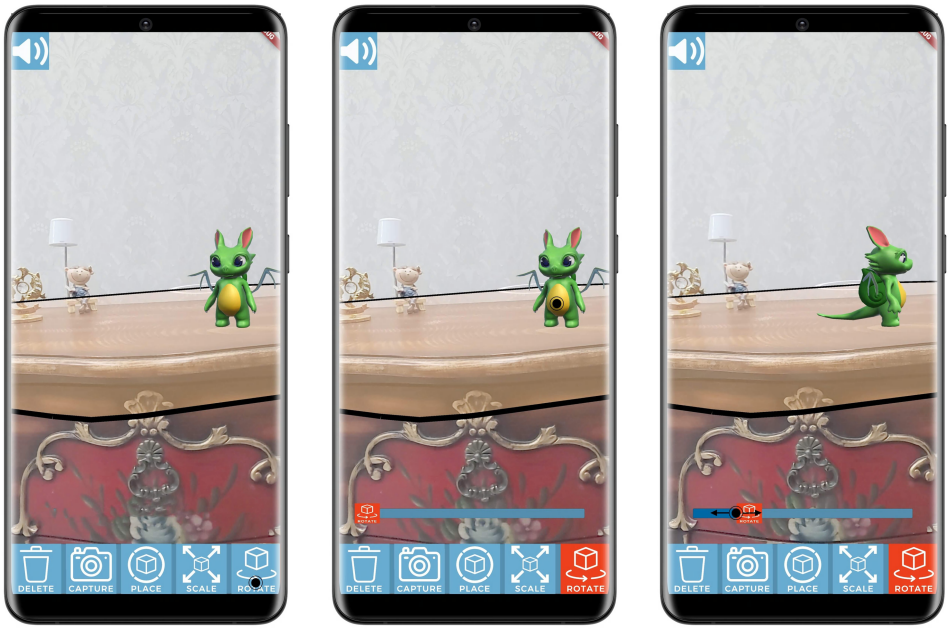
(ii) Object rotated right

Figure 6.9: Feature: Rotate using Gestures

Rotate using Slider

The steps to rotate an object using a slider are

1. The user presses the 'Rotate' button to view the rotation slider (Figure 6.10i)
2. The user selects an object by clicking on it (Figure 6.10ii)
3. The user slides through the slider to alter the angle of the object (Figure 6.10iii)



(i) Press rotate button to show slider

(ii) Slide through the slider

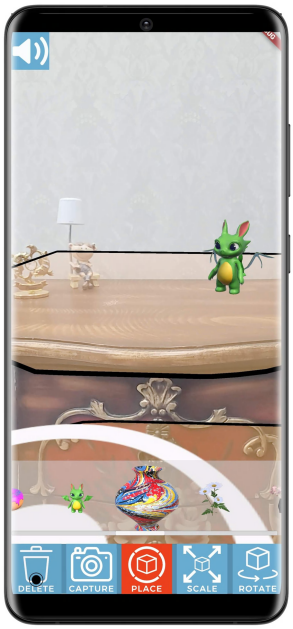
(iii) Object angle changes

Figure 6.10: AR Interface - Iteration One

6.1.7 Delete Object

The user selects an object (Figure 6.11i) and presses the delete button (Figure 6.11ii) to delete a placed object from the scene.

(i) Select an object by tapping on it

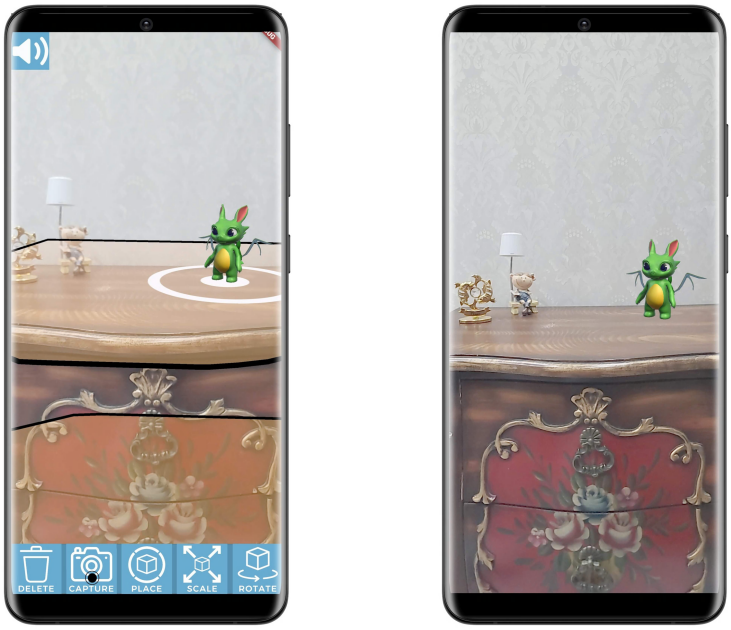


(ii) Delete button pressed to remove object

Figure 6.11: Feature: Delete Object

6.1.8 Capture Scene

Once the user is satisfied with the curated scene, the user presses the 'Capture' button (Figure 6.12i) to take a screen snapshot which is saved to the gallery (Figure 6.12ii).



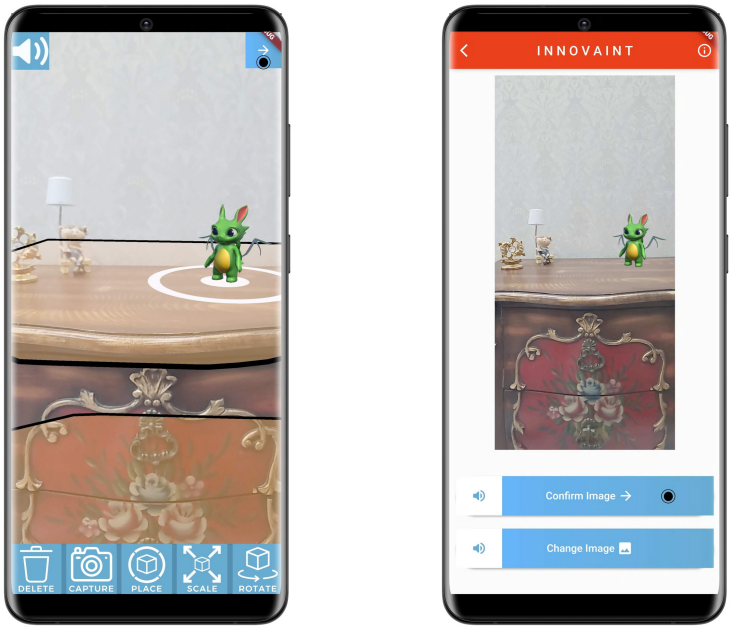
(i) Press capture button to take snapshot

(ii) Image saved to gallery

Figure 6.12: Feature: Capture Scene

Capture Multiple Scenes

The user could take multiple screenshots while curating the scene and press the next button (Figure 6.13i) once finalized to move to the AST interface (Figure 6.13ii).



(i) Press next button after finalizing

(ii) Proceeds to AST interface

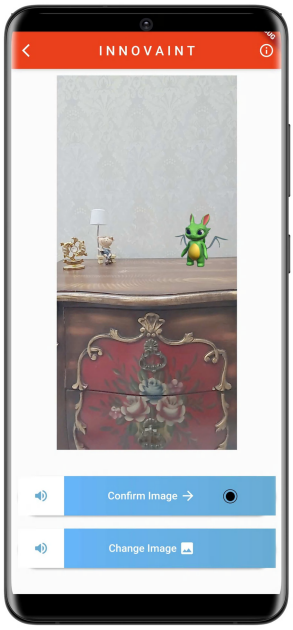
Figure 6.13: Feature: Capture Multiple Scenes

6.1.9 Choose Content Image

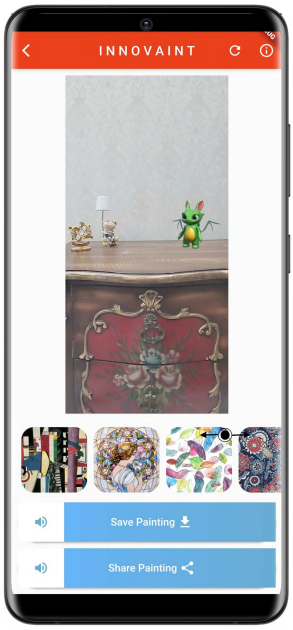
The user gets an option to select an image to stylize

Confirm Content Image

The last captured image is selected by default to stylize. The user could select this default image (Figure 6.14i) to stylize (Figure 6.14ii).



(i) Press confirm button to select default image

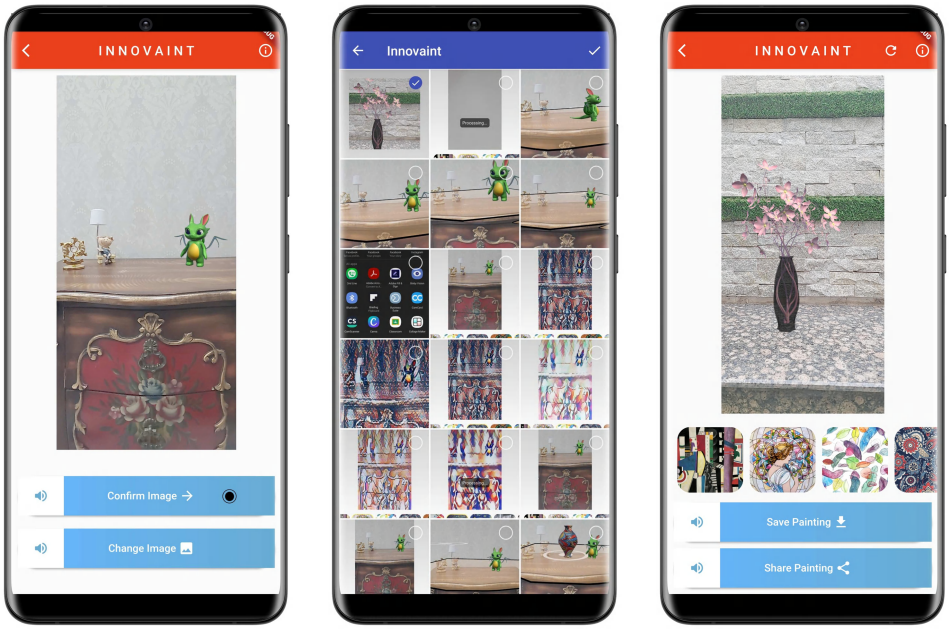


(ii) Default image selected

Figure 6.14: Feature: Confirm Content Image

Change Image

The user could change the default image (Figure 6.15i) and pick any other image (Figure 6.15ii) taken while curating the scene from the gallery to stylize it (Figure 6.15iii).



(i) Press change button to change image

(ii) Select image from gallery

(iii) Content image to stylize updated

Figure 6.15: Feature: Change Content Image

6.1.10 Browse Painting Styles

The user could browse through the different painting styles to apply to the chosen image (Figure 6.16).

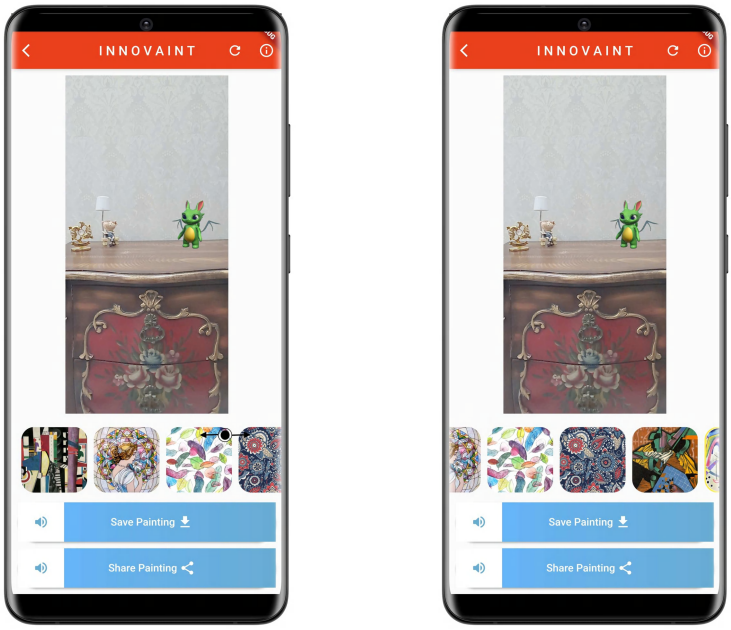
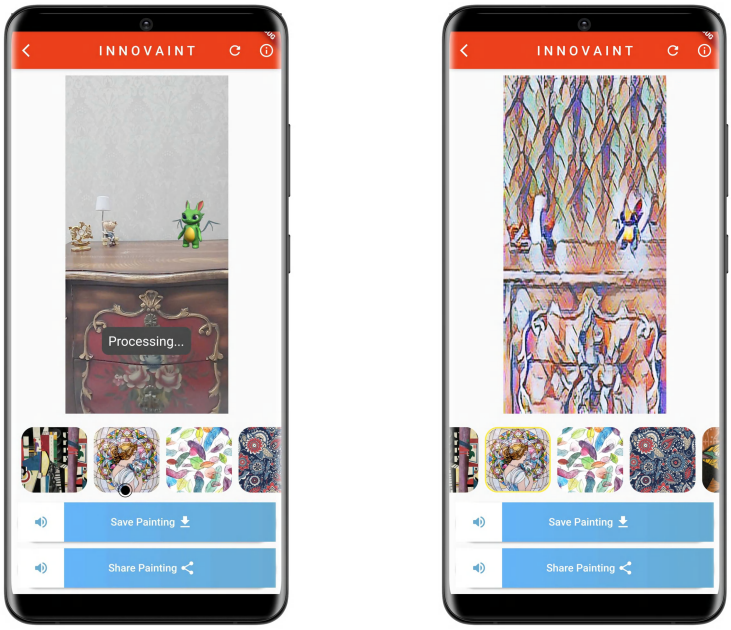


Figure 6.16: Feature: Browse Painting Styles

6.1.11 Generate Painting

The user selects a painting style filter (Figure 6.17i) which gets applied to the image to generate a painting (Figure 6.17ii).



(i) Select style filter to apply style

(ii) Generates Painting

Figure 6.17: Feature: Generate Painting

Generate Multiple Paintings

The user could keep generating more paintings by selecting the different style filters (Figure 6.18).

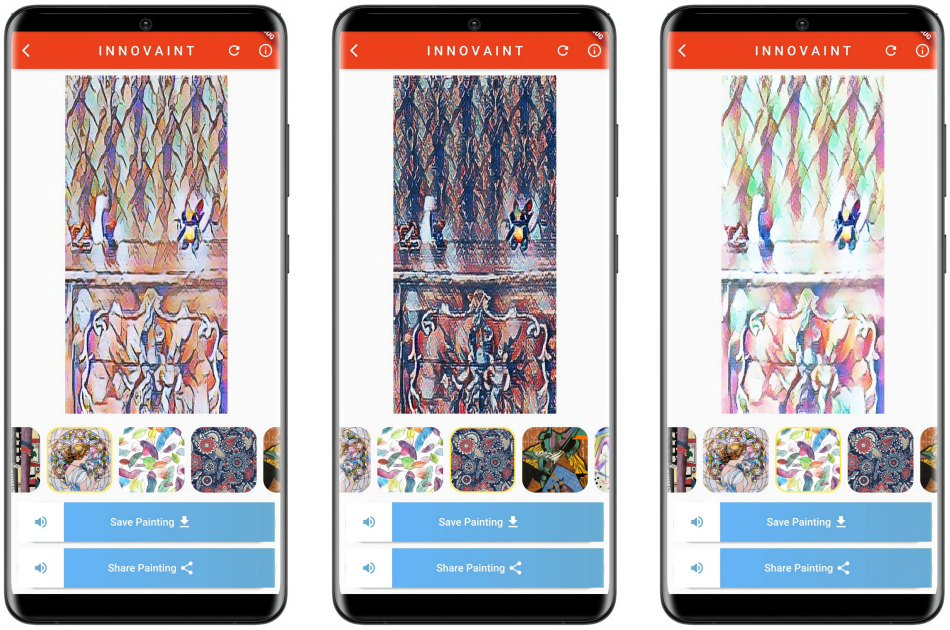
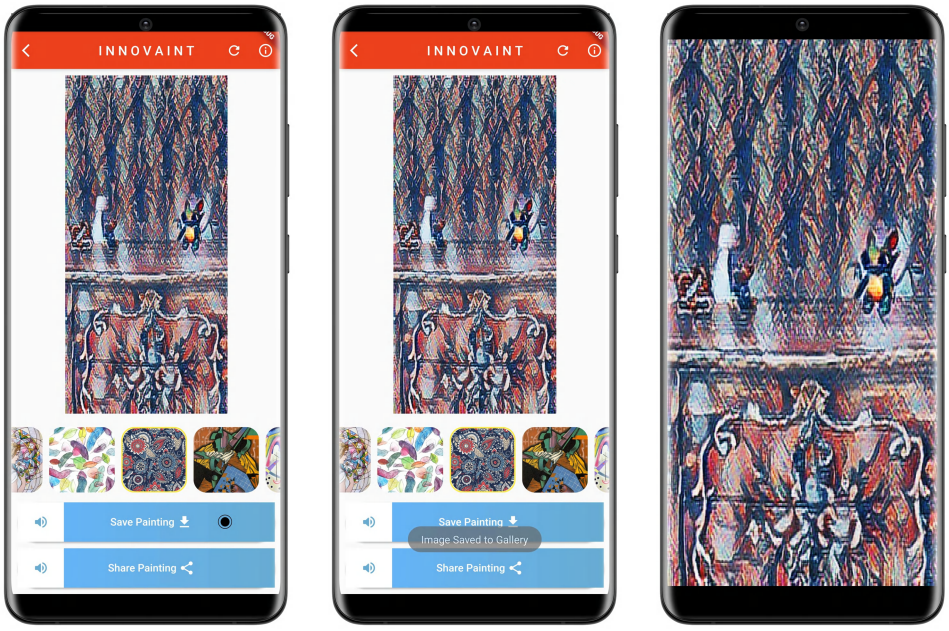


Figure 6.18: Feature: Generate Multiple Paintings

6.1.12 Save Painting

The user presses the 'Save Painting' button (Figure 6.19i) to save the generated painting to the gallery (Figure 6.19ii).



(i) Press save button

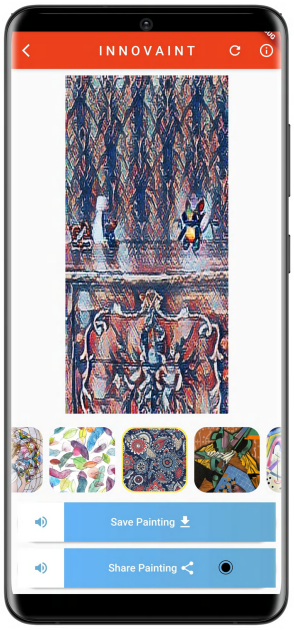
(ii) Textual and Auditory feedback

(iii) Painting saved to gallery

Figure 6.19: Feature: Save Painting

6.1.13 Share Painting

The user presses the 'Share Painting' button (Figure 6.20i) to share the generated painting externally to third-party services (Figure 6.20ii).



(i) Press share button to view options

(ii) External sharing options displayed

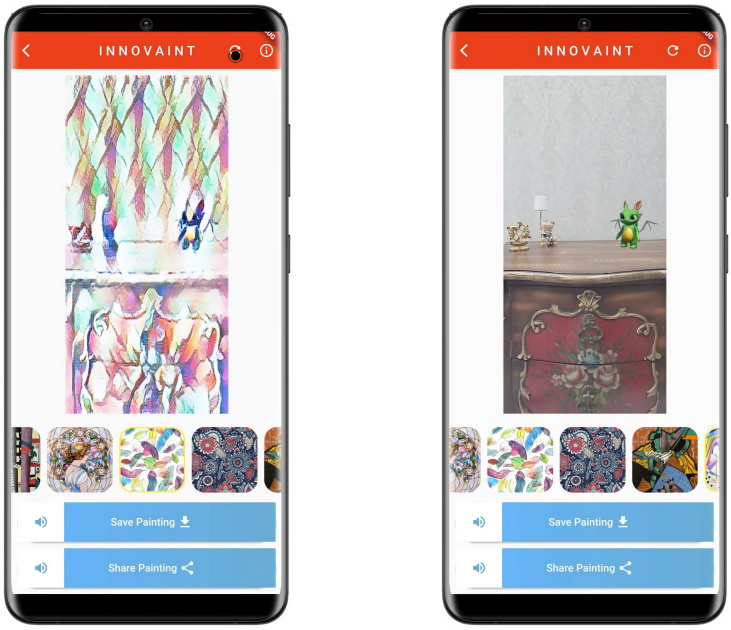
Figure 6.20: Feature: Share Painting

6.2 Additional Features

This section details the additional features included in the application to support people with aphasia

6.2.1 Revert to Original Content Image

The user presses the 'Revert' button icon (Figure 6.21i) to restore the original content image (Figure 6.21ii).



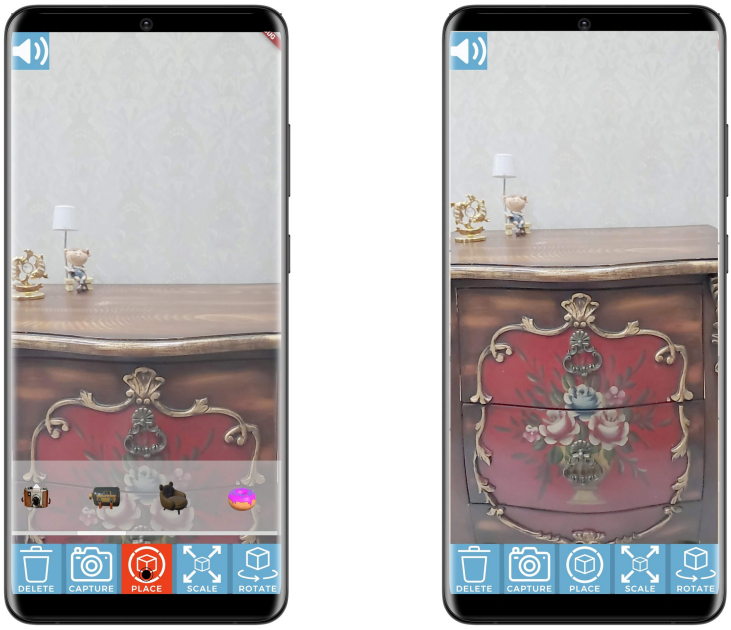
(i) Press revert button icon

(ii) Content image restores

Figure 6.21: Feature: Revert to Original Content Image

6.2.2 Toggle Interface Elements

The user could toggle elements to show (Figure 6.22i) or hide (Figure 6.22ii) them to allow for maximum space to play with the scene.



(i) TTS buttons' panel shows

(ii) TTS buttons' panel hides

Figure 6.22: Additional Feature: Toggle UI Elements

6.2.3 Text-to-Speech

Augmented Reality Interface

The user presses the speaker button to display the TTS buttons' panel (Figure 6.23i). Then, the user presses the TTS button below the respective action buttons to read aloud the text (Figure 6.23ii).

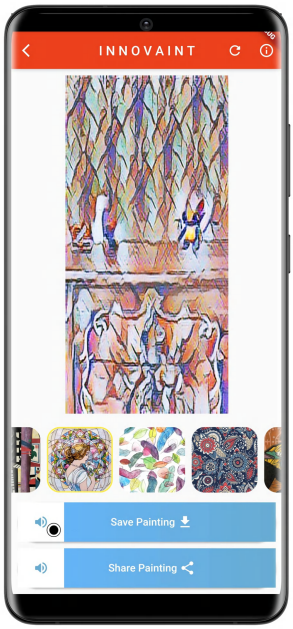
(i) Press speaker button to show panel

(ii) Press speaker button on panel to hear label

Figure 6.23: Additional Feature: TTS for AR Interface

Artistic Style Transfer Interface

The user presses the speaker button next to the action button to hear the text on the button (Figure 6.24).



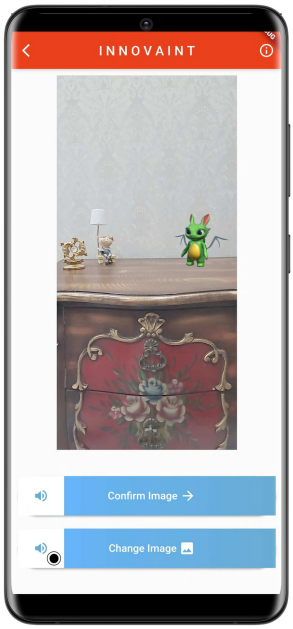


Figure 6.24: Additional Feature: TTS for AST Interface

Home Screen

The user presses the speaker button in the app bar to listen to all the displayed text on the screen (Figure 6.25).

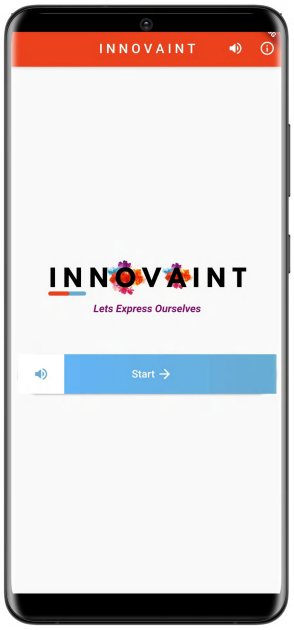


Figure 6.25: Additional Feature: TTS for Home Screen

6.2.4 Help Section

The user presses the help icon from any screen to access the help section. The user interactions on the help screen (Figure 6.26) include:

- Press the 'Play' button to play or pause the video
- Press the 'Mute' button to mute or unmute the video
- Press the 'Full Screen' button to watch the video in full-screen

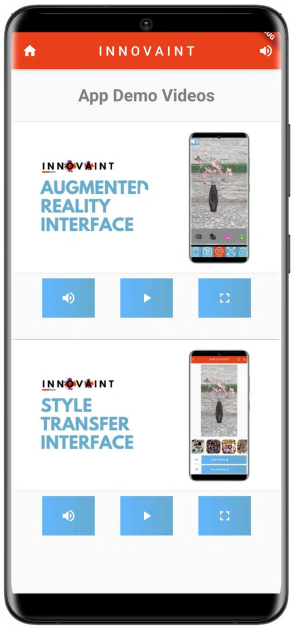


Figure 6.26: Additional Feature: Help Section

6.3 Application Screenshots

6.3.1 Smartphone Device

Figure 6.27 presents screenshots taken on a smartphone with a display resolution of 1440x3200 pixels.

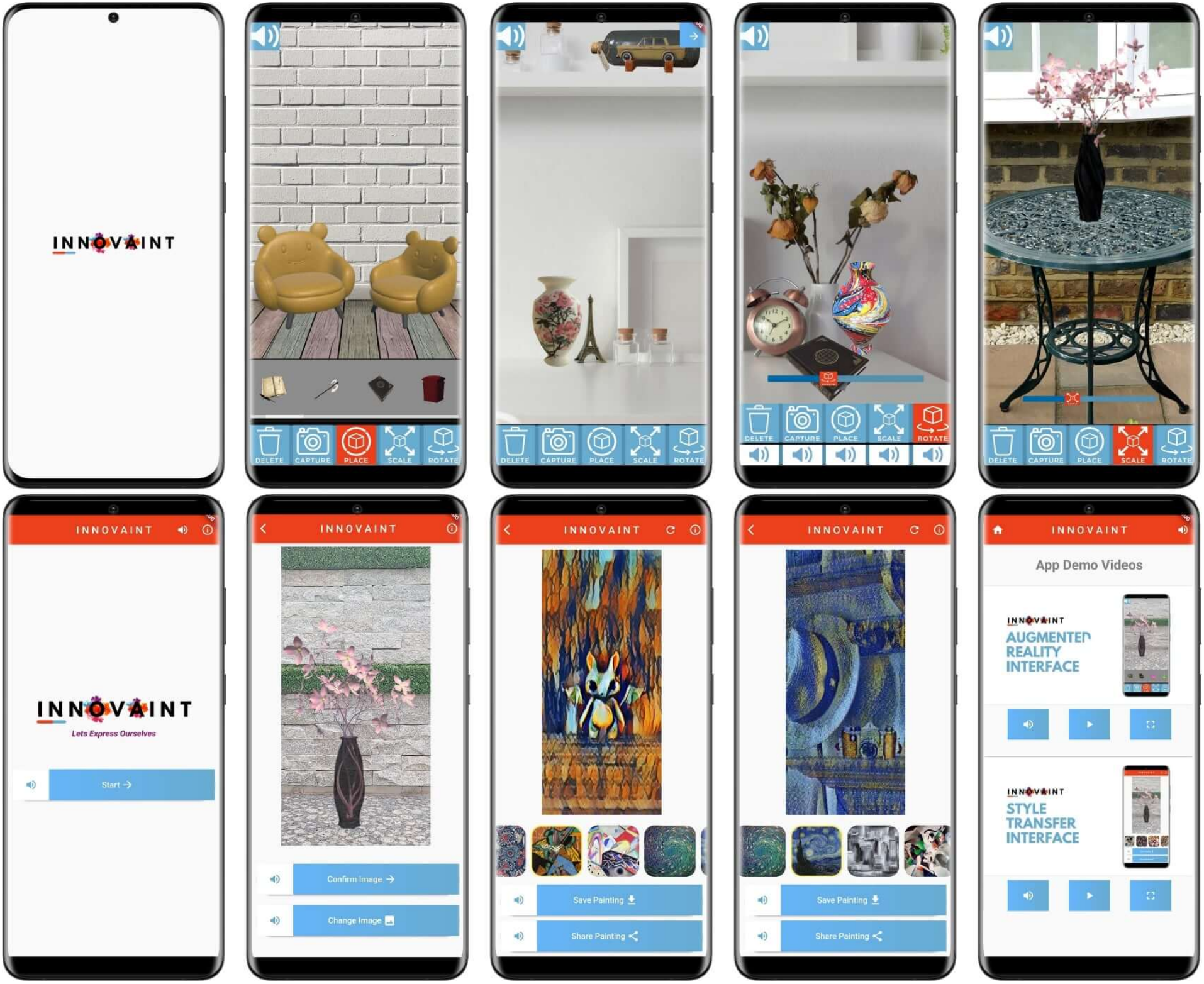


Figure 6.27: Application Screenshots from a Smartphone

6.3.2 Tablet Device

Figure 6.28 presents screenshots taken on a tablet with a display resolution of 1200x2000 pixels.

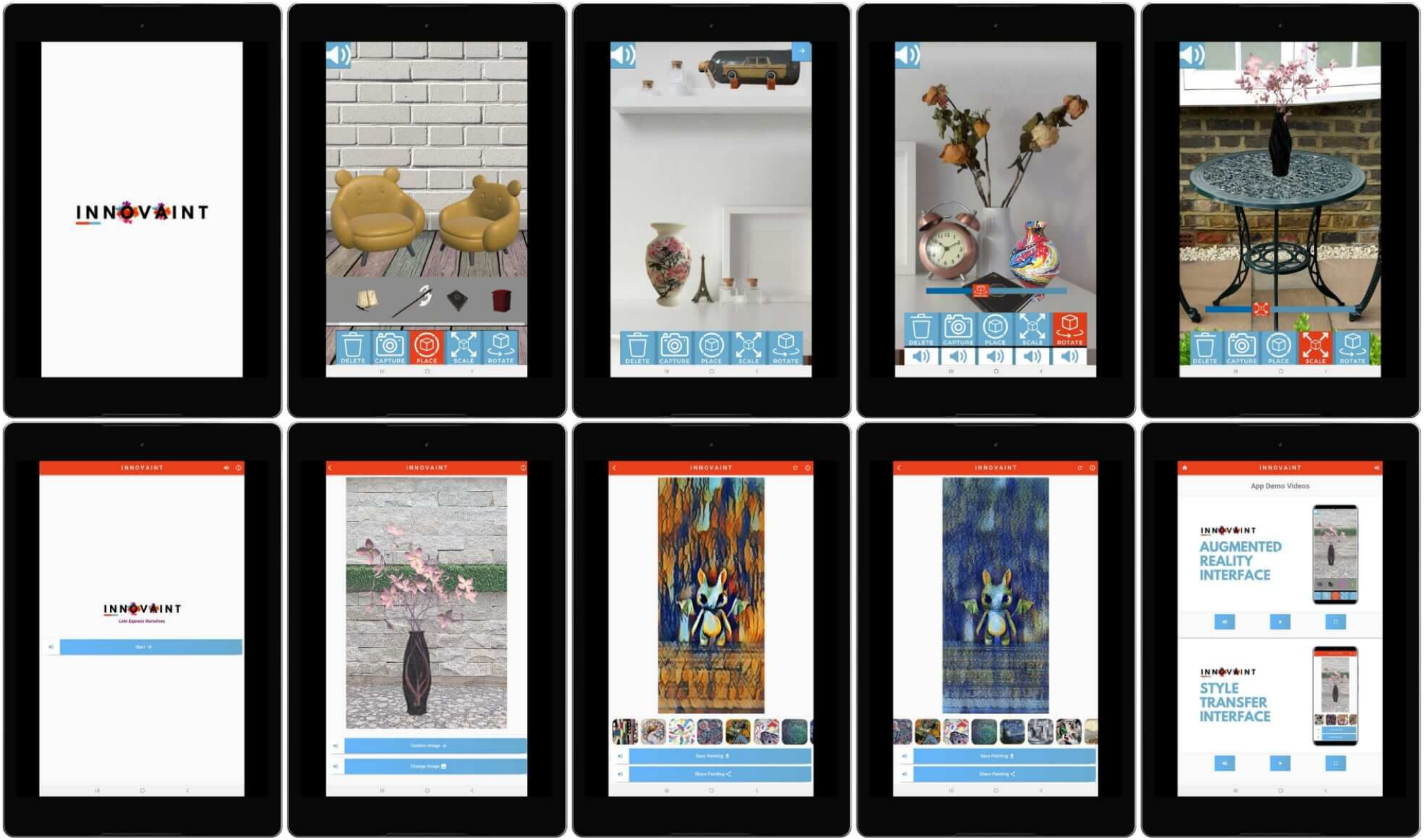


Figure 6.28: Application Screenshots from a Tablet

Chapter 7

Technical Evaluation

7.1 Functional Testing

The application functionality was manually tested against all the functional requirements (Section 3.6), and the results for the main requirements are reported as test cases in this section. The test cases for the remaining requirements are detailed in Appendix E. To support people with aphasia, all the functions require either a single step or two steps.

7.1.1 Augmented Reality Interface Test Cases

1. Requirement: The application shall allow the user to place an object in the scene (Table 7.1)

2. Requirement: The application shall allow the user to delete a placed object (Table 7.2)

3. Requirement: The application shall allow the user to scale an object (Table 7.3)

4. Requirement: The application shall allow the user to rotate an object (Table 7.4)

5. Requirement: The application shall allow the user to drag an object across the plane (Table 7.5)

6. Requirement: The application shall allow the user to capture the created augmented scene (Table 7.6)

Test Case ID	AR04	**Test Case Name:**	**Place Object**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to place an object in the environment			
Pre-Condition	Object is selected from the objects' scrollview			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User taps on the detected plane while the object is selected from the objects' scrollview	Application should instantiate an object on the position touched	As Expected	**Pass**
	User taps on the detected plane while the object is not selected from the objects' scrollview	Application should not instantiate an object	As Expected	**Pass**
	User drags the crosshair on the screen	Application should update the position of the crosshair on the screen	As Expected	**Pass**
2.	User taps on the crosshair while the object is selected from the objects' scrollview	Application should instantiate an object on the position touched	As Expected	**Pass**
	User taps on the crosshair while the object is not selected from the objects' scrollview	Application should not instantiate an object	As Expected	**Pass**
Post-Condition	The object is placed in the environment			

Table 7.1: Test Case for Placing Object

Test Case No	AR07	**Test Case Name:**	**Delete Object**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to delete a placed object.			
Pre-Condition	The object should be placed.			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User taps on an object to select it	Application should save the reference of the selected object	As Expected	**Pass**
2.	User presses the 'Delete' button while object is selected	Application should delete the object	As Expected	**Pass**
	User presses the 'Delete' button while object is not selected	Application should not do anything	As Expected	**Pass**
Post-Condition	The object has been removed from the scene			

Table 7.2: Test Case for Deleting Object

Test Case ID	AR08	**Test Case Name:**	**Scale Object**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to scale the object.			
Pre-Condition	The object should be placed.			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User selects an object	Application should save the reference of the selected object	As Expected	**Pass**
2.	User scales an object using gestures while object is selected	Application should scale the object according to the pinch size	As Expected	**Pass**
	User scales an object using gestures while object is not selected	Application should not change the size	As Expected	**Pass**
	User scales an object using slider while object is selected	Application should scale the object according to the slider value	As Expected	**Pass**
	User scales an object using slider while object is not selected	Application should not change the size	As Expected	**Pass**
Post-Condition	The object size has increased or decreased.			

Table 7.3: Test Case for Scaling Object

Test Case ID	AR09	**Test Case Name:**	**Rotate Object**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to rotate the object.			
Pre-Condition	The object should be placed.			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User taps on an object to select it	Application should save the reference of the selected object	As Expected	**Pass**
2.	User rotates an object using gestures while object is selected	Application should rotate the object according to the turn gesture	As Expected	**Pass**
	User rotates an object using gestures while object is not selected	Application should not do anything	As Expected	**Pass**
	User rotates an object using slider while object is selected	Application should rotate the object according to the slider value	As Expected	**Pass**
	User rotates an object using slider while object is not selected	Application should not do anything	As Expected	**Pass**
Post-Condition	The object angle has changed			

Table 7.4: Test Case for Rotating Object

Test Case No	AR10	**Test Case Name:**	**Drag Object**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to drag the object.			
Pre-Condition	The object should be placed.			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User taps on an object to select it	Application should save the reference of the selected object	As Expected	**Pass**
2.	User drags an object by sliding the finger while object is selected	Application should change the position of the placed object	As Expected	**Pass**
	User drags an object by sliding the finger while object is not selected	Application should not do anything	As Expected	**Pass**
Post-Condition	The object position has changed.			

Table 7.5: Test Case for Dragging Object

Test Case No	AR11	**Test Case Name:**	**Capture Screen**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to take a snapshot of the curated scene			
Pre-Condition	None			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User presses the 'Capture' button	Application should take a screenshot	As Expected	**Pass**
		Application should save the image to gallery	As Expected	**Pass**
2.	User presses the 'Next' button to finalize the image	Application should proceed to the style transfer screen	As Expected	**Pass**
Post-Condition	The style transfer screen is displayed			

Table 7.6: Test Case for Capturing Screen

Test Case No	AST01	**Test Case Name:**	**Change Content Image**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to select a content image for style transfer			
Pre-Condition	The user has proceeded to the image screen			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User presses 'Change Image' button	Application should open the gallery for image selection	As Expected	**Pass**
2.	User selects an image from the gallery	Application should change the content image	As Expected	**Pass**
	User selects an invalid image from the gallery	Application should not select that as the content image	As Expected	**Pass**
Post-Condition	The content image has changed			

Table 7.7: Test Case for Changing Content Image

7.1.2 Artistic Style Transfer Interface Test Cases

7. Requirement: The application shall allow the user to select a content image for style transfer (Table 7.7)

8. Requirement: The application shall allow the user to apply a painting style to the image (Table 7.8)

9. Requirement: The application shall allow the user to save the generated painting (Table 7.9)

10. Requirement: The application shall allow the user to share the generated painting (Table 7.10)

Test Case No	AST04	**Test Case Name:**	**Apply Style Transfer**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to apply a style transfer			
Pre-Condition	The user has finalized a style image			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User selects a style filter by tapping on an image	Application should merge the style image with the content image	As Expected	**Pass**
Post-Condition	The painting is generated			

Table 7.8: Test Case for Applying Style Transfer

Test Case No	AST06	**Test Case Name:**	**Save Painting**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to save the painting to gallery			
Pre-Condition	The user has generated a painting			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User presses the 'Save Painting' button	Application should save the painting to gallery	Painting Saved to Gallery	**Pass**
		Application should provide confirmation of saving	Text and Speech Feedback confirming save	**Pass**
	User presses the 'Save Painting' button and painting is not generated	Application should prompt to apply style transfer first	Text and Speech Feedback prompting to apply style	**Pass**
Post-Condition	The painting is saved to gallery			

Table 7.9: Test Case for Saving Painting

Test Case No	AST07	**Test Case Name:**	**Share Painting**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to share the painting			
Pre-Condition	The user has generated a painting			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User presses the 'Share Painting' button	Application should display the external sharing options	As Expected	**Pass**
	User presses the 'Share Painting' button and painting is not generated	Application should prompt to apply style transfer first	Text and Speech Feedback prompting to apply style	**Pass**
2.	User selects the WhatsApp icon, and there is an internet connection	Application should share the painting on WhatsApp as an image	Painting Shared Externally	**Pass**
	User selects the Whatsapp icon, and there is no internet connection	Whatsapp should display connection error message. "Check your internet connection and try again."	Painting Not Shared	**Pass**
Post-Condition	The painting is shared as an image to Whatsapp			

Table 7.10: Test Case for Sharing Painting

7.2 Performance Testing

Flutter profiling was used to test the performance of the application in a profile mode. The CPU and UI rendering times for a typical user session were recorded across different devices, and this section reports the results.

The session compromised of the following steps

1. Placing an object in the environment
2. Scaling the placed object
3. Capturing the scene
4. Selecting the default image
5. Applying a painting style to the image
6. Saving the generated painting

Flutter measures application performance on two main threads - UI thread and Raster thread [74]. The UI thread builds the widget tree by executing the dart code for building screens. The raster thread runs on the CPU and is responsible for rendering this widget tree by communicating with the GPU to display screens.

The application was tested on four android devices, including a tablet device. All the devices were running an android version between Android 9 and 11. The devices used and their specifications are listed in table 7.11.

ID	Device	OS	Chipset	CPU	GPU	RAM
N20	Samsung Galaxy Note 20 Ultra	Android 11	Snapdragon 865+	3.0, 2.42, 1.8 GHz octa-core	Adreno 650	12GB
S10	Samsung Galaxy S10 Plus	Android 10	Exynos 9820	2.73, 2.31, 1.95 GHz octa-core	Mali-G76 MP12	8GB
N9	Samsung Galaxy Note 9	Android 10	Exynos 9810	2.8, 1.7 GHz octa-core	Mali-G72 MP18	6GB
TS6	Samsung Galaxy Tab S6 Lite	Android 11	Exynos 9611	2.3, 1.7 GHz octa-core	Mali-G72 MP3	4GB

Table 7.11: Devices used for Performance Testing

The profiling granularity was set to medium, meaning that the CPU collected samples at a rate of 1 sample every 250 microseconds. The average frame rate for rendering in the raster thread is 7.2 milliseconds(ms)/frame across the four tested devices. The maximum rendering time is 22.6 ms/frame. For the UI thread, the average loading frame rate is 1.5 ms/frame, and the maximum is 13.6 ms/frame across all the tested devices. An average of 60 frames are loaded per second on all the devices. Figure 7.1 summarizes the maximum rendering results, and figure 7.2 summarizes the average rendering results across all the devices.

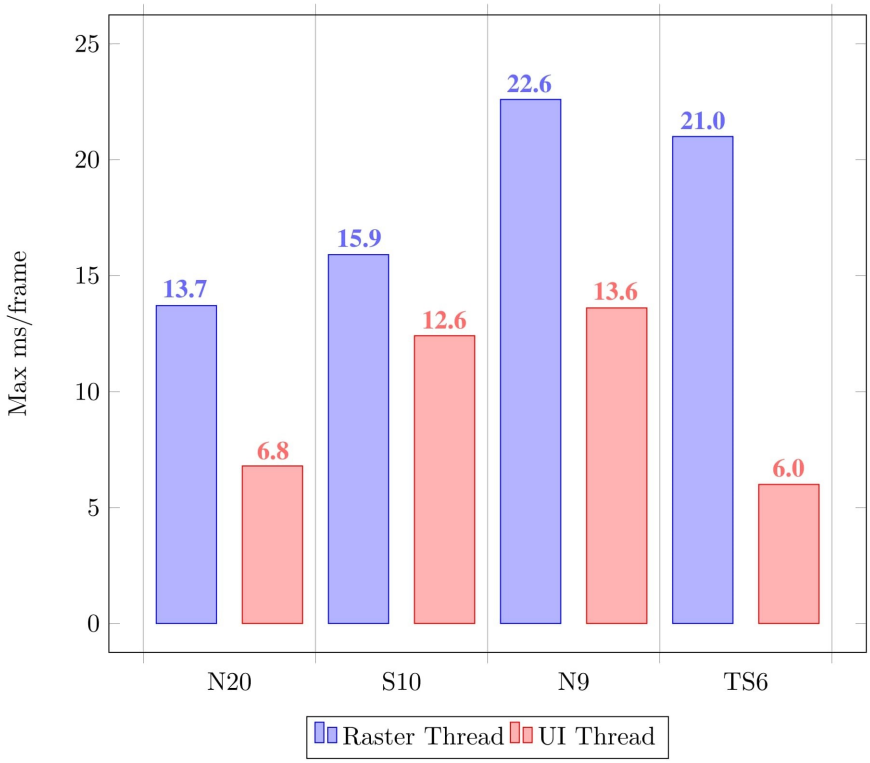


Figure 7.1: Maximum Frame Rendering Time for Threads across Devices

The overall application performance is satisfactory as it runs smoothly for the most part through a session, with frame rates considerably below the standard of 16 ms per frame yielding 60 FPS. Jank is primarily detected on the AR interface setup and style transfer screen. There is a spike in frame rendering times while setting up the unity player for the AR interface as the unity project has to be loaded and connected to the application. Another spike is observed on the style transfer screen while loading painting style images due to the I/O processing required.

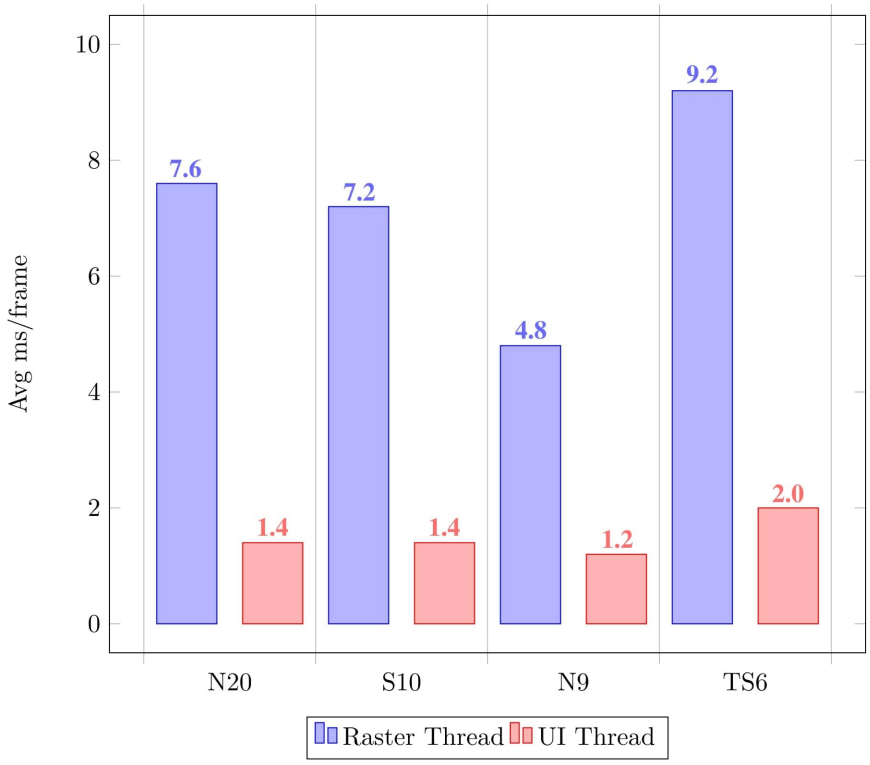


Figure 7.2: Average Frame Rendering Time for Threads across Devices

Chapter 8

Legal, Social, Ethical and Professional Issues

8.1 Ethical Issues

The rights of the subjects should be carefully taken into account when a research investigation uses humans as participants in that study [75]. Hence, as this study included the involvement of human participants to act as proxies for feedback on the application, ethical clearance was required to proceed. A minimal risk ethical clearance to conduct this research was submitted, and proxies were approached after the approval from the ethics board. A copy of the ethical approval is attached in Appendix **??**.

The interviews with the proxies were held after informing them about the purpose of the study and the data processing details. Also, the interviews were proceeded only after formal written consent (Refer to Appendix H to see the Information Sheet and Sample Consent Form). The anonymity and the confidentiality of the participants were maintained throughout the study.

8.2 Legal Issues

The data was processed under UK data protection law (including the UK General Data Protection Regulation (UK GDPR) and the Data Protection Act 2018). Participants were promised not to be named or identifiable in the final report and throughout the project and only be presented pseudonymously in the report. The data was only retained for the period of

the research and erased after that. Secure, encrypted online storage was used to store the data collected. Data was only be shared within the research team and did not include any involvement of third parties. Furthermore, no data was shared outside of the EU, except as part of the report.

8.3 Professional Issues

Integrity has been maintained throughout the project by strictly following the Code of Conduct and Code of Good Practice guidelines set out by the British Computer Society [76]. This project uses various open-source libraries and plugins, and all such instances have been explicitly stated. Any ideas, design inspiration, image credits used have also been appropriately referenced. This guarantees due regard for intellectual property, removing any false claims of ownership.

Further, performance and security concerns have been considered in developing the application, making sure to pose no potential harm to both the user and the device running it. Relevant user permissions have also been included in the application when access to phone storage and the camera device is needed. Lastly, all third-party software used is certified as safe to use and has been thoroughly tested by the respective development teams.

8.4 Social Issues

Considering this application has been developed for vulnerable groups, safety and ease of use have been pivotal. The design process ensured maximizing accessibility. These considerations were further verified and enhanced, and usability for the target group was confirmed by expert proxies.

Moreover, the application presents extensive potential benefits to the user. The focus on improving communication and expression for people with aphasia could improve their lifestyle through the betterment of interaction and stimulating discussions. Finally, testing the prototype for iOS devices and publishing it on the Google Play Store would benefit a larger audience.

Chapter 9

Conclusion and Future Work

9.1 Conclusion

Communication forms the basis of all human social interactions and relationships. Hence, it is vital to support people facing communicative challenges through technology or otherwise. Existing solutions offer restrictive platforms for self-expression, whereas this study proposes a creative form of expression employing communication and art therapy techniques.

This project presents a novel AAC tool to aid expression for people with aphasia through visual arts using Augmented Reality and Artistic Style Transfer technologies. Based on a qualitative analysis of proxy evaluations, it can be concluded that the application has the potential to be used in communication therapy by providing an alternative communicative medium. The results establish simple usability of the application by people with aphasia due to the accessible design and feature support.

Innovaint promotes self-expression by allowing scene curations and painting generations that convey emotions, feelings, stories, or personal interests, among others. The composition created provides a point for discussion and stimulates conversation. Conversely, Innovaint could be used playfully to produce digital artworks.

Furthermore, this perspective of incorporating AR and AST techniques in developing accessible technology is promising. It offers potential for further exploration by employing these to design technology for people with diverse needs.

9.2 Future Work

The project could be extended by performing user studies with actual people with aphasia to evaluate the application and making the required design or feature changes accordingly. Furthermore, user studies would provide valuable insights on how someone with aphasia interacts with the application, and it would help to reiterate the design process.

The application development could also be extended to include more virtual objects and painting styles. Painting styles could be made more flexible to allow the user more control over applying the style by using sliders to increase or decrease the effect. The artistic style transfer model could also be customized to experiment with different parameters and possibly allow users more control over how a particular image is stylized.

Another possible extension could be taking user mood input into the application and suggest objects and style filters accordingly, which would aid communication therapy and help express themselves better.

Finally, the application could also be deployed on iOS to support more devices. The application could also be published to the Google Playstore and Apple App Store to make it available to a broader audience who would benefit from it.

9.3 Limitations

Although real people with aphasia could not be involved in this study, proxies who have experience of closely working with them participated. They provided constructive feedback on the application design and covered up the unavailability of people with aphasia in evaluating the application. Further, a video conference mode was adapted to replace the physical face-to-face sessions with the therapists, which was equally valuable.

The application was developed to run on Android versions greater than nine and not any older version because of an essential plugin to integrate the augmented reality interface into flutter. However, as this plugin was not compatible with older versions, the application could not support them.

Moreover, good lighting conditions are required for the augmented reality interface to correctly detect planes which is a limitation for working with AR in general. Lastly, if a user has severe mobility issues due to their health condition, they might not take complete advantage of the application. In this scenario, however, the phone could be set on a tripod and used effectively.

9.4 Learning Outcomes

This project proved to be a valuable knowledge addition enhancing both technical and soft skills. The process provided significant insights into factors and considerations that influence designing accessible technology. Moreover, the discussions with specialists were enlightening and allowed a deeper understanding of this domain and how people with aphasia interact with technology. It also taught the ethical implications and the process when conducting research involving human participants. Moreover, the technical skills gained by working with Augmented Reality interfaces and Machine Learning models are unparalleled. The course of this project increased the expertise in developing AR applications and style transfer techniques. Although it had been challenging to dive into a new domain and technical area, the results have been fulfilling. Lastly, the gained knowledge is sure to provide support in all future endeavours.

References

[1] *FurnitureAR*. [Online]. Available: https://github.com/Shubhra22/FurnitureAR (visited on 07/16/2021).

[2] *Placing and Manipulating Objects in AR*. [Online]. Available: https://learn.unity.com/tutorial/placing-and-manipulating-objects-in-ar (visited on 07/16/2021).

[3] *Rotate and Scale AR Object using Sliders - Unity 2020*, en. [Online]. Available: https://www.youtube.com/watch?v=43XRkmAmscM (visited on 07/16/2021).

[4] *Aphasia Definitions*, en-US. [Online]. Available: https://www.aphasia.org/aphasia-definitions/ (visited on 08/01/2021).

[5] *Aphasia FAQs*, en-US. [Online]. Available: https://www.aphasia.org/aphasia-faqs/ (visited on 09/01/2021).

[6] D. Edwards, *Art Therapy*, en. SAGE, Sep. 2004, ISBN: 9780761947516.

[7] U. Technologies, *Unity as a Library. For native mobile apps written in Objective C & Java — AR for Android & iOS — Unity*, en. [Online]. Available: https://unity.com/features/unity-as-a-library (visited on 08/28/2021).

[8] N. A. Association, *National aphasia association*, en, 2021. [Online]. Available: https://www.aphasia.org/.

[9] J. J. Hinckley and M. E. Packard, "Family education seminars and social functioning of adults with chronic aphasia," en, *Journal of Communication Disorders*, vol. 34, no. 3, pp. 241–254, May 2001, ISSN: 00219924. DOI: 10.1016/S0021-9924(01)00049-1. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0021992401000491 (visited on 09/06/2021).

[10] W. Sandt-Koenderman, "Aphasia rehabilitation and the role of computer technology: Can we keep up with modern times?" en, *International journal of speech-language pathology*, vol. 13, no. 1, pp. 21–27, 2011.

[11] Lingraphica, *Smalltalk free aphasia communication and video apps — lingraphica*, en, 2021. [Online]. Available: https://www.aphasia.com/smalltalk-aphasia-apps/.

[12] M. Garcia, "A speech therapy game application for aphasia patient neurorehabilitation–a pilot study of an mhealth app," en, *International Journal of Simulation: Systems, Science Technology*, vol. 20, pp. 5–1, 2019.

[13] T. Neate, A. Roper, S. Wilson, and J. Marshall, "Empowering expression for users with aphasia through constrained creativity," en, *CHI '19 Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, vol. 650, no.), S. Brewster and G. Fitzpatrick, Eds., 0000–0001–6950–6294, 2019.

[14] C. Tamburro, T. Neate, A. Roper, and S. Wilson, "Accessible Creativity with a Comic Spin," in *The 22nd International ACM SIGACCESS Conference on Computers and Accessibility*, ser. ASSETS '20, Virtual Event, Greece: Association for Computing Machinery, Oct. 2020, pp. 1–11, ISBN: 9781450371032. DOI: 10.1145/3373625.3417012. [Online]. Available: https://doi.org/10.1145/3373625.3417012 (visited on 09/04/2021).

[15] T. Neate, A. Roper, S. Wilson, J. Marshall, and M. Cruice, "Creatable content and tangible interaction in aphasia," en, in *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, 2020.

[16] *Aphasia Therapy Guide*, en-US. [Online]. Available: https://www.aphasia.org/aphasia-resources/aphasia-therapy-guide/ (visited on 07/04/2021).

[17] N. Simmons-Mackie and M. Schultz, "The role of humour in therapy for aphasia," *Aphasiology*, vol. 17, no. 8, pp. 751–766, Jan. 2003, ISSN: 0268-7038. DOI: 10.1080/02687030344000229. [Online]. Available: https://doi.org/10.1080/02687030344000229 (visited on 09/05/2021).

[18] A. Nykänen, H. Nyrkkö, M. Nykänen, R. Brunou, and P. Rautakoski, "Communication therapy for people with aphasia and their partners (APPUTE)," *Aphasiology*, vol. 27, no. 10, pp. 1159–1179, Oct. 2013, ISSN: 0268-7038. DOI: 10.1080/02687038.2013.802284.

[Online]. Available: https://doi.org/10.1080/02687038.2013.802284 (visited on 09/05/2021).

[19] *PACE Therapy*, en-US. [Online]. Available: https://www.aphasia.com/aphasia-resource-library/aphasia-treatments/pace/ (visited on 09/07/2021).

[20] J. Parrish, "Art and Aphasia: A Literary Review and Exhibition," Ph.D. dissertation, 2014. [Online]. Available: https://scholarworks.wmich.edu/honors_theses/2445.

[21] O. Johnston, "Helping the Client Confronted with Aphasia Improve his Communication Skills through the Use of Art Therapy," en, *Canadian Art Therapy Association Journal*, vol. 13, no. 1-2, pp. 27–42, Sep. 1999, ISSN: 0832-2473, 2377-360X. DOI: 10.1080/08322473.1999.11432242. [Online]. Available: http://www.tandfonline.com/doi/full/10.1080/08322473.1999.11432242 (visited on 09/05/2021).

[22] *About Art Therapy*, en-US. [Online]. Available: https://arttherapy.org/about-art-therapy/ (visited on 08/07/2021).

[23] S.-H. Kim, M.-Y. Kim, J.-H. Lee, and S.-i. Chun, "Art Therapy Outcomes in the Rehabilitation Treatment of a Stroke Patient: A Case Report," en, *Art Therapy: Journal of the American Art Therapy Association*, vol. 25, no. 3, pp. 129–133, 2008, ISSN: 0742-1656. [Online]. Available: https://eric.ed.gov/?id=EJ811585 (visited on 09/05/2021).

[24] N. Helm-Estabrooks, M. L. Albert, and N. Helm-Estabrooks, *Manual of aphasia and aphasia therapy*, 2nd ed. Austin, Tex: Pro-Ed, 2004, ISBN: 9780890799635.

[25] Statista, *Topic: Augmented reality (ar*, en, 2021. [Online]. Available: https://www.statista.com/topics/3286/augmented-reality-ar/.

[26] Leftroniccom, *29 + augmented reality statistics [updated 2021*, en, 2021. [Online]. Available: https://leftronic.com/blog/augmented-reality-statistics/.

[27] C. Arth, R. Grasset, L. Gruber, T. Langlotz, A. Mulloni, and D. Wagner, *The history of mobile augmented reality*, en, arXiv preprint arXiv:1505.01319.', 2015.

[28] B. Thomas, B. Close, J. Donoghue, J. Squires, P. De Bondi, M. Morris, and W. Piekarski, "Arquake: An outdoor/indoor augmented reality first person application," en, in *Digest of Papers. Fourth International Symposium on Wearable Computers*, 2000.

[29] O. Bimber and R. Raskar, *Spatial augmented reality*, en, A. Peters, Ed., 2005.

[30] P. Cooke, *The digital deluge: 4.0 industry from ai to houzz*, en.

[31] C. Alves, J. Reis, and February, "The intention to use e-commerce using augmented reality-the case of ikea place," en, in *International Conference on Information Technology Systems*, Cham: Springer, 2020, pp. 114–123.

[32] W. Brush, *World brush*, en, 2021. [Online]. Available: https://worldbrush.net.

[33] V. Media, *Figment ar app — viro media*, fr, 2021. [Online]. Available: https://viromedia.com/figment.

[34] A. Critic, *Holo app review - holograms in augmented reality (ios*, en, 2021. [Online]. Available: https://arcritic.com/1041/holo-app-review/.

[35] *The Future of Mobile AR: A Comparison Between ARCore, ARKit, AR Foundation, and Vuforia*, en. [Online]. Available: https://www.credera.com/insights/the-future-of-mobile-ar-a-comparison-between-arcore-arkit-ar-foundation-and-vuforia (visited on 09/20/2021).

[36] *5 Best Augmented Reality Development Tools [2021 Guide]*, en. [Online]. Available: https://program-ace.com/blog/augmented-reality-sdk/ (visited on 08/20/2021).

[37] L. A. Gatys, A. S. Ecker, and M. Bethge, "A Neural Algorithm of Artistic Style," en, Aug. 2015. [Online]. Available: https://arxiv.org/abs/1508.06576v2 (visited on 09/05/2021).

[38] Y. Jing, Y. Yang, Z. Feng, J. Ye, Y. Yu, and M. Song, "Neural Style Transfer: A Review," en, May 2017. [Online]. Available: https://arxiv.org/abs/1705.04058v7 (visited on 09/06/2021).

[39] Y. Li, N. Wang, J. Liu, and X. Hou, "Demystifying Neural Style Transfer," en, in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, Melbourne, Australia: International Joint Conferences on Artificial Intelligence Organization, Aug. 2017, pp. 2230–2236, ISBN: 9780999241103. DOI: 10.24963/ijcai.2017/310. [Online]. Available: https://www.ijcai.org/proceedings/2017/310 (visited on 09/06/2021).

[40] C. Li and M. Wand, "Combining Markov Random Fields and Convolutional Neural Networks for Image Synthesis," en, Jan. 2016. [Online]. Available: https://arxiv.org/abs/1601.04589v1 (visited on 09/06/2021).

[41] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual Losses for Real-Time Style Transfer and Super-Resolution," en, Mar. 2016. [Online]. Available: https://arxiv.org/abs/1603.08155v1 (visited on 09/06/2021).

[42] V. Dumoulin, J. Shlens, and M. Kudlur, "A Learned Representation For Artistic Style," en, Oct. 2016. [Online]. Available: https://arxiv.org/abs/1610.07629v5 (visited on 09/06/2021).

[43] *TensorFlow Hub*. [Online]. Available: https://tfhub.dev/google/magenta/arbitrary-image-stylization-v1-256/2 (visited on 08/05/2021).

[44] G. Ghiasi, H. Lee, M. Kudlur, V. Dumoulin, and J. Shlens, "Exploring the structure of a real-time, arbitrary neural artistic stylization network," en, May 2017. [Online]. Available: https://arxiv.org/abs/1705.06830v2 (visited on 09/06/2021).

[45] Deepart.io, *Deepart.io - become a digital artist*, it, 2021. [Online]. Available: https://deepart.io.

[46] Pikazoappcom, *Pikazo*, fr, 2021. [Online]. Available: https://www.pikazoapp.com/.

[47] lucidco, fr, 2021. [Online]. Available: https://www..

[48] Prisma-aicom, *Prisma labs*, fr, 2021. [Online]. Available: https://prisma-ai.com/.

[49] Artistomycom, *Artisto app*, en, 2021. [Online]. Available: https://artisto.my.com/.

[50] A. Al Mahmud, "Considerations for designing technology with and for persons with aphasia," in *Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: the Future of Design*, ser. OzCHI '14, Sydney, New South Wales, Australia: Association for Computing Machinery, Dec. 2014, pp. 535–538, ISBN: 9781450306539. DOI: 10.1145/2686612.2686698. [Online]. Available: https://doi.org/10.1145/2686612.2686698 (visited on 09/04/2021).

[51] *Language-Light UX – Nine Interaction Design Guidelines for Aphasia*, en-GB. [Online]. Available: http://languagelightux.org/ (visited on 08/27/2021).

[52] I. F. Lee, *Joyful: The Surprising Power of Ordinary Things to Create Extraordinary Happiness*, en. Ebury Publishing, Sep. 2018, Google-Books-ID: gB2_tAEACAAJ, ISBN: 9781846045448.

[53] O. Blazhenkova and M. M. Kumar, "Angular Versus Curved Shapes: Correspondences and Emotional Processing," en, *Perception*, vol. 47, no. 1, pp. 67–89, Jan. 2018, ISSN: 0301-0066. DOI: 10.1177/0301006617731048. [Online]. Available: https://doi.org/10.1177/0301006617731048 (visited on 09/05/2021).

[54] L. Howarth, "Drawing Clocks without Time: Alzheimer's, Marginalization, and the Liberatory Potential of Object Memoir," *Proceedings of the Annual Conference of CAIS / Actes du congrès annuel de l'ACSI*, Dec. 1969, ISSN: 2562-7589. DOI: 10.29173/cais886. [Online]. Available: https://journals.library.ualberta.ca/ojs.cais-acsi.ca/index.php/cais-asci/article/view/886 (visited on 09/05/2021).

[55] M. Woudstra, A. Al Mahmud, and J.-B. Martens, "A snapshot diary to support conversational storytelling for persons with aphasia," in *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, ser. MobileHCI '11, Stockholm, Sweden: Association for Computing Machinery, Aug. 2011, pp. 641–646, ISBN: 9781450305419. DOI: 10.1145/2037373.2037474. [Online]. Available: https://doi.org/10.1145/2037373.2037474 (visited on 09/04/2021).

[56] E. Daemen, P. Dadlani, J. Du, Y. Li, P. Erik-Paker, J.-B. Martens, and B. de Ruyter, "Designing a Free Style, Indirect, and Interactive Storytelling Application for People with Aphasia," en, in *Human-Computer Interaction – INTERACT 2007*, C. Baranauskas, P. Palanque, J. Abascal, and S. D. J. Barbosa, Eds., ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2007, pp. 221–234, ISBN: 9783540747963. DOI: 10.1007/978-3-540-74796-3_21.

[57] "The association between colors and emotions for emotional words and facial expressions," en, *Color Research & Application*, vol. 43, no. 4, pp. 624–624, Aug. 2018, ISSN: 03612317. DOI: 10.1002/col.22236. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/col.22236 (visited on 09/06/2021).

[58] C. H. Moon, "Theorizing Materiality in Art Therapy : Negotiated Meanings," in *Materials & Media in Art Therapy*.

[59] I. Isewon, J. Oyelade, and O. Oladipupo, "Design and Implementation of Text To Speech Conversion for Visually Impaired People," *International Journal of Applied Information Systems*, vol. 7, no. 2, pp. 25–30, Apr. 2014, ISSN: 22490868. DOI: 10.5120/ijais14-

451143. [Online]. Available: http://research.ijais.org/volume7/number2/ijais14-451143.pdf (visited on 09/06/2021).

[60] *Speech synthesis*, en, Page Version ID: 1041597004, Aug. 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Speech_synthesis&oldid=1041597004 (visited on 09/05/2021).

[61] *Progress in speech synthesis*, eng. New York : Springer, 1997, ISBN: 9780387947013. [Online]. Available: http://archive.org/details/progressinspeech0000unse (visited on 09/05/2021).

[62] T. Levin, B. M. Scott, B. Borders, K. Hart, J. Lee, and A. Decanini, "Aphasia Talks: Photography as a means of communication, self-expression, and empowerment in persons with aphasia," eng, *Topics in Stroke Rehabilitation*, vol. 14, no. 1, pp. 72–84, Feb. 2007, ISSN: 1074-9357. DOI: 10.1310/tsr1401-72.

[63] *APACHE LICENSE, VERSION 2.0*, en. [Online]. Available: https://www.apache.org/licenses/LICENSE-2.0 (visited on 09/06/2021).

[64] *Magenta*, en. [Online]. Available: https://magenta.tensorflow.org/ (visited on 08/18/2021).

[65] *MobileNetV2*, en. [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/keras/applications/mobilenet_v2/MobileNetV2 (visited on 08/20/2021).

[66] *Artistic Style Transfer with TensorFlow Lite*, en. [Online]. Available: https://www.tensorflow.org/lite/examples/style_transfer/overview (visited on 09/06/2021).

[67] O. Hasler, S. Blaser, and S. Nebiker, "IMPLEMENTATION AND FIRST EVALUATION OF AN INDOOR MAPPING APPLICATION USING SMARTPHONES AND AR FRAMEWORKS," en, *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-2/W17, pp. 135–141, Nov. 2019, ISSN: 2194-9034. DOI: 10.5194/isprs-archives-XLII-2-W17-135-2019. [Online]. Available: https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLII-2-W17/135/2019/ (visited on 09/16/2021).

[68] J. L. Boyd-Graber, S. S. Nikolova, K. A. Moffatt, K. C. Kin, J. Y. Lee, L. W. Mackey, M. M. Tremaine, and M. M. Klawe, "Participatory design with proxies: Developing a desktop-PDA system to support people with aphasia," in *Proceedings of the SIGCHI*

Conference on Human Factors in Computing Systems, New York, NY, USA: Association for Computing Machinery, Apr. 2006, pp. 151–160, ISBN: 9781595933720. [Online]. Available: https://doi.org/10.1145/1124772.1124797 (visited on 09/05/2021).

[69] L. S. Nowell, J. M. Norris, D. E. White, and N. J. Moules, "Thematic Analysis: Striving to Meet the Trustworthiness Criteria," en, *International Journal of Qualitative Methods*, vol. 16, no. 1, p. 1 609 406 917 733 847, Dec. 2017, ISSN: 1609-4069. DOI: 10.1177/1609406917733847. [Online]. Available: https://doi.org/10.1177/1609406917733847 (visited on 09/06/2021).

[70] K. Hagood, T. Moore, T. Pierre, P. Messamer, G. Ramsberger, and C. Lewis, "Naming practice for people with aphasia in a mobile web application: Early user experience," in *Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility*, ser. ASSETS '10, Orlando, Florida, USA: Association for Computing Machinery, Oct. 2010, pp. 273–274, ISBN: 9781605588810. DOI: 10.1145/1878803.1878867. [Online]. Available: https://doi.org/10.1145/1878803.1878867 (visited on 09/05/2021).

[71] R. Shenoy, S. Nayak, M. K. Hegde, N. Kini, P. P. Kundapur, and G. Krishnan, "Development of an android application in kannada to enhance picture naming skills in persons with aphasia," in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Sep. 2017, pp. 2134–2140. DOI: 10.1109/ICACCI.2017.8126161.

[72] T. Neate, A. Roper, and S. Wilson, "Painting a Picture of Accessible Digital Art," en, Athens, Greece, Jul. 2020. DOI: 10.1145/3373625.3418019. [Online]. Available: https://doi.org/10.1145/3373625.3418019 (visited on 09/06/2021).

[73] M. G. Obiorah, A. M. Piper, and M. Horn, "Independent Word Discovery for People with Aphasia," in *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*, ser. ASSETS '17, Baltimore, Maryland, USA: Association for Computing Machinery, Oct. 2017, pp. 325–326, ISBN: 9781450349260. DOI: 10.1145/3132525.3134790. [Online]. Available: https://doi.org/10.1145/3132525.3134790 (visited on 09/05/2021).

[74] *Flutter performance profiling*, en. [Online]. Available: https://flutter.dev/docs/perf/rendering/ui-performance (visited on 09/12/2021).

[75] D. F. Polit and B. P. Hungler, *Nursing research: principles and methods*, 6th ed. Philadelphia: Lippincott, 1999, ISBN: 9780781715621.

[76] *BCS Code of Conduct — BCS*. [Online]. Available: https://www.bcs.org/membership/become-a-member/bcs-code-of-conduct/ (visited on 09/08/2021).

Appendix A

Plugins and Packages

A.0.1 Unity

- Flutter Unity Widget - https://github.com/juicycleff/flutter-unity-view-widget
- LeanTouch - https://assetstore.unity.com/packages/tools/input-management/lean-touch-30111
- Native Gallery - https://github.com/yasirkula/UnityNativeGallery

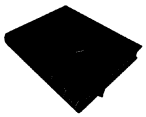
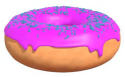
A.0.2 Flutter

- Flutter Unity Widget - https://pub.dev/packages/flutter_unity_widget
- Flutter Bloc - https://pub.dev/packages/flutter_bloc
- Freezed Annotation - https://pub.dev/packages/freezed_annotation
- Image - https://pub.dev/packages/image
- Multi Image Picker - https://pub.dev/packages/multi_image_picker
- TF Lite - https://pub.dev/packages/tflite_flutter
- Image Gallery Saver - https://pub.dev/packages/image_gallery_saver
- Share Plus - https://pub.dev/packages/share_plus
- Flutter Toast - https://pub.dev/packages/fluttertoast
- YouTube Player - https://pub.dev/packages/youtube_player_flutter

- Flutter TTS - https://pub.dev/packages/flutter_tts
- Build Runner - https://pub.dev/packages/build_runner
- Native Splash - https://pub.dev/packages/flutter_native_splash

Appendix B

Virtual 3D Objects

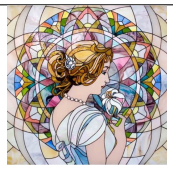
Name	Preview	Source
Antique Book		https://bit.ly/38KjCSF
Axe		https://bit.ly/3yM6LKO
Book		https://bit.ly/3DNSqk1
Bottled Car		https://bit.ly/2YjNYt4
Chair		https://bit.ly/3yM6G9a
Donut		https://bit.ly/2WQtVSy
Dragon		https://bit.ly/3jLZAh2
Ebru Vase		https://bit.ly/3yVXuzv
Fountain Pen		https://bit.ly/3h3ckOQ
Gardening Trowel		https://bit.ly/2WVDijy

Name	Preview	Source
Japanese Vase		https://bit.ly/3tgaiiC
Keyboard		https://bit.ly/3DKVjCi
Mask		https://bit.ly/2YuwZ7B
Pink Flower		https://bit.ly/3kXLQif
Pink Vase		https://bit.ly/3DRKvm6
Phone Booth		https://bit.ly/3yLQUeP
Screwdriver		https://bit.ly/3jGuG9E
Vintage Camera		https://bit.ly/3yRfURW
White Flower		https://bit.ly/38GIhrd
Wooden Spoon		https://bit.ly/2WTbtZC

Table B.1: Virtual Object Sources

Appendix C

Painting Style Filters

Style No.	Preview	Source
Style 1		https://bit.ly/3DQ0xgO
Style 2		https://bit.ly/3jKNhS4
Style 3		https://bit.ly/3n2Uoap
Style 4		https://bit.ly/3zLtRSF
Style 5		https://bit.ly/3yLUANF
Style 6		https://bit.ly/3jJIDUg
Style 7		https://bit.ly/2WNysFL
Style 8		https://bit.ly/3zL1ofF
Style 9		https://bit.ly/3DQ0ReI
Style 10		https://bit.ly/2WVHfVq

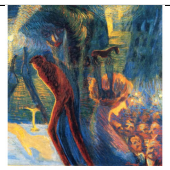
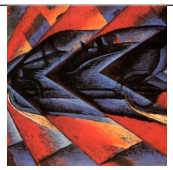
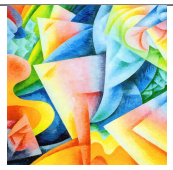
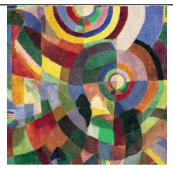
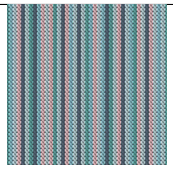
Style No.	Preview	Source
Style 11		https://bit.ly/3h5EwjL
Style 12		https://bit.ly/3yW1lfX
Style 13		https://bit.ly/3kPN7YO
Style 14		https://bit.ly/3kSaUHK
Style 15		https://bit.ly/3BGGyP5
Style 16		https://bit.ly/3kVgVmT
Style 17		https://bit.ly/3zNNsRY
Style 18		https://bit.ly/3zZIy4J
Style 19		https://bit.ly/2WVmO6K
Style 20		https://bit.ly/3kTMM7w

Table C.1: Painting Style Filters' Sources

Appendix D

Interview Transcripts

D.1 Formative Evaluations

D.1.1 First Proxy Interview Transcript

Appendix E

Additional Test Cases

E.0.1 Augmented Reality Interface Test Cases

1. Requirement: The application shall allow the user to browse through objects (Table E.1)

Test Case ID	AR01	**Test Case Name**	**Browse Objects**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to browse through the objects.			
Pre-Condition	None			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User presses the 'Place' button	Application should display the object's scrollview	As Expected	**Pass**
2.	User slides through the scroll view to view objects	Application should allow the user to scroll through all the objects in the scrollview	As Expected	**Pass**
Post-Condition	The objects have been displayed			

Table E.1: Test Case for Browsing Objects

2. Requirement: The application shall allow the user to select an object from the scroll view (Table E.2)

Test Case ID	AR02	**Test Case Name:**	**Select Object**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to select an object from the scrollview.			
Pre-Condition	Object's scrollview displayed			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User taps on an image in the scrollview to select it	Application should save an object reference corresponding to the image	As Expected	**Pass**
Post-Condition	The object gets selected			

Table E.2: Test Case for Selecting Objects

Test Case ID	AR03	**Test Case Name:**	**Detect Plane**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to detect a horizontal plane			
Pre-Condition	None			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User hovers over a horizontal surface	Application should detect the plane	As Expected	**Pass**
		Application should show the detected plane	As Expected	**Pass**
	User points camera at a horizontal surface	Application should detect the plane	As Expected	**Pass**
		Application should show the detected plane	As Expected	**Pass**
Post-Condition	The horizontal planes get detected			

Table E.3: Test Case for Detecting Plane

3. Requirement: The application shall allow the user to detect a plane for placing objects (Table E.3)

Test Case ID	AR05	**Test Case Name:**	**Place Multiple Objects**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to place multiple objects in the environment			
Pre-Condition	An object is already placed in the environment			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User taps on the detected plane while the object is selected from the objects' scrollview	Application should instantiate an object on the position touched	As Expected	**Pass**
	User taps on the detected plane while the object is not selected from the objects' scrollview	Application should not instantiate an object	As Expected	**Pass**
	User drags the crosshair on the screen	Application should update the position of the crosshair on the screen	As Expected	**Pass**
2.	User taps on the crosshair while the object is selected from the objects' scrollview	Application should instantiate an object on the position touched	As Expected	**Pass**
	User taps on the crosshair while the object is not selected from the objects' scrollview	Application should not instantiate an object	As Expected	**Pass**
Post-Condition	Multiple objects are placed in the environment			

Table E.4: Test Case for Placing Multiple Objects

4. Requirement: The application shall allow the user to place multiple objects in the scene (Table E.4)

5. Requirement: The application shall allow the user to select a placed object (Table E.5)

6. Requirement: The application shall allow the user to toggle user interface elements (Table E.6)

Test Case ID	AR06	**Test Case Name:**	**Select Placed Object**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to select a placed object in the environment			
Pre-Condition	Object Is placed in the environment			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User taps on a placed object to select it	Application should save reference of the object tapped on	As Expected	**Pass**
Post-Condition	The object gets selected			

Table E.5: Test Case for Selecting Placed Object

Test Case No	AR12	**Test Case Name:**	**Toggle UI Elements**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to toggle all the user interface elements			
Pre-Condition	None			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User presses the 'Place' button and the objects' scrollview is displayed	Application should hide the objects' scrollview	As Expected	**Pass**
	User presses the 'Place' button and the objects' scrollview is hidden	Application should display the objects' scrollview	As Expected	**Pass**
	User presses the 'Scale' button and the scale slider is displayed	Application should hide the scale slider	As Expected	**Pass**
	User presses the 'Scale' button and the scale slider is hidden	Application should display the scale slider	As Expected	**Pass**
	User presses the ''Rotate' button and the rotate slider is displayed	Application should hide the rotate slider	As Expected	**Pass**
	User presses the 'Rotate' button and the scale slider is hidden	Application should display the rotate slider	As Expected	**Pass**
	User presses the 'Speaker' button and the TTS buttons' panel is displayed	Application should hide the TTS buttons' panel	As Expected	**Pass**
	User presses the 'Speaker' button and the TTS buttons' panel is hidden	Application should display the TTS buttons' panel	As Expected	**Pass**
Post-Condition	The UI element is either hidden or displayed			

Table E.6: Test Case for Toggling UI Elements

Test Case No	AST02	**Test Case Name:**	**Select Content Image**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to view the last captured image			
Pre-Condition	The user has proceeded to the style transfer screen			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User presses 'Confirm Image' button after viewing the default content image	Application should use the default image as the content image	As Expected	**Pass**
Post-Condition	The content image has been finalized			

Table E.7: Test Case for Selecting Content Image

Test Case No	AST03	**Test Case Name:**	**Browse Style Filters**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to browse through painting style filters			
Pre-Condition	The user has finalized a content image			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User slides through the list view to view painting styles	Application should allow the user to scroll through all the paintings in the listview	As Expected	**Pass**
Post-Condition	The painting styles are displayed			

Table E.8: Test Case for Browsing Style Filters

E.0.2 Artistic Style Transfer Interface Test Cases

7. Requirement: The application shall allow the user to select the last captured image (Table E.7)

8. Requirement: The application shall allow the user to browse through painting styles (Table E.8)

Test Case No	AST05	**Test Case Name:**	**View Generated Painting**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to view the generated painting			
Pre-Condition	The user has applied the style transfer			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User selects a style filter by tapping on an image	Application should merge the style image with the content image	As Expected	**Pass**
		Application should display the painting in an image view	As Expected	**Pass**
Post-Condition	The painting is displayed			

Table E.9: Test Case for Viewing Generated Painting

Test Case No	AST08	**Test Case Name:**	**Revert to Original Image**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to revert to the original curated image			
Pre-Condition	The user has generated a painting			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User presses the 'Revert' button	Application should remove the style filter applied	**As Expected**	**Pass**
		Application should display the original image	**As Expected**	**Pass**
Post-Condition	The original image is displayed			

Table E.10: Test Case for Reverting to Original Image

9. Requirement: The application shall allow the user to view the generated painting (Table E.9)

10. Requirement: The application shall allow the user to revert to the original image (Table E.10)

Test Case No	AST09	**Test Case Name:**	**Auditory Feedback**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to receive auditory feedback on save			
Pre-Condition	The user has generated a painting			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User presses the 'Save Painting' button	Application should save the painting to gallery	As Expected	**Pass**
		Application should read-aloud save confirmation	As Expected	**Pass**
	User presses the 'Save Painting' button and painting is not generated	Application should read-aloud prompt to apply style	As Expected	**Pass**
Post-Condition	The feedback is read out loud			

Table E.11: Test Case for Receiving Auditory Feedback

11. Requirement: The application shall allow the user to receive auditory feedback on saving the image (Table E.11)

Test Case No	G01	**Test Case Name:**	**Read-aloud Text**	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to read-aloud the displayed text			
Pre-Condition	None			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User presses the 'Speaker' button next to call for actions	Application should read-aloud the text on the corresponding button	As Expected	**Pass**
	User presses the 'Speaker' button in the appbar	Application should read-aloud all displayed text on the screen	As Expected	**Pass**
Post-Condition	The text is read aloud			

Table E.12: Test Case for Reading-Aloud text

E.0.3 General Application Test Cases

12. Requirement: The application shall allow the user to listen to any text displayed on screen (Table E.12)

13. Requirement: The application shall allow the user to access the help section (Table **??**)

14. Requirement: The application shall allow the user to interact with the demo videos (Table E.14)

Test Case No	G02	Test Case Name:	Access Help Section	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to access the help section			
Pre-Condition	None			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User presses the 'Info' icon button in the app bar from the AST interface	Application should navigate to the Help Section screen	As Expected	**Pass**
	User presses the 'Info' icon button in the app bar from the Home Screen	Application should navigate to the Help Section screen	As Expected	**Pass**
Post-Condition	The Help Screen is displayed			

Table E.13: Test Case for Accessing Help Section

Test Case No	G03	Test Case Name:	Video Interactions	
Tested By	anonymous	**Tested On**	01/09/2021	
Description	This tests if the user is able to interact with the demo videos in the help section			
Pre-Condition	The help screen is displayed			
Step No	**Test Steps**	**Expected Result**	**Actual Result**	**Status**
1.	User presses the 'Play' button	Application should play the video	As Expected	**Pass**
	User presses the 'Mute' button	Application should mute the video sound	As Expected	**Pass**
	User presses the 'Full Screen' button	Application should display the video in full screen mode	As Expected	**Pass**
2.	User presses the 'Pause' button	Application should pause the video	As Expected	**Pass**
	User presses the 'Unmute' button	Application should mute the video sound	As Expected	**Pass**
Post-Condition	None			

Table E.14: Test Case for Demo Video Interactions

Appendix F

Performance Testing Charts

Charts from performance overlays from within the application and from dart developer tools displaying the raster and the UI thread rendering times are presented here for each individual device.

Samsung Galaxy Note 20 Ultra

Refer to figure F.1 and F.5.

Samsung Galaxy S10 Plus

Refer to figure F.2 and F.6.

Samsung Galaxy Note 9

Refer to figure F.3 and F.7.

Samsung Galaxy Tab S6 Lite

Refer to figure F.4 and F.8.

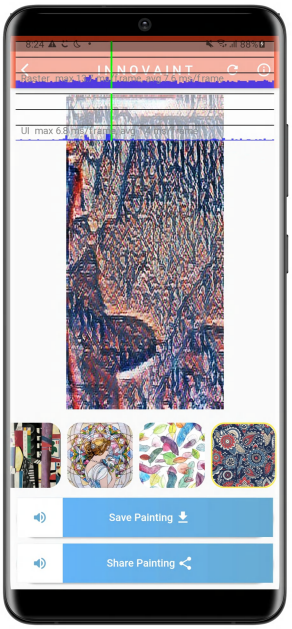


Figure F.1: Performance Overlay View - Note 20 Ultra

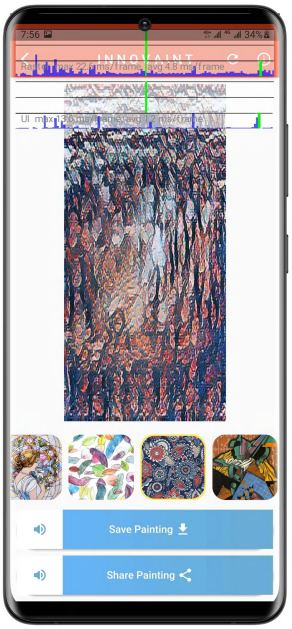


Figure F.2: Performance Overlay View - S10 plus

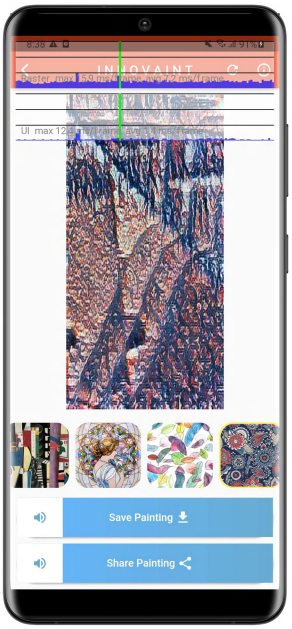


Figure F.3: Performance Overlay View - Note 9

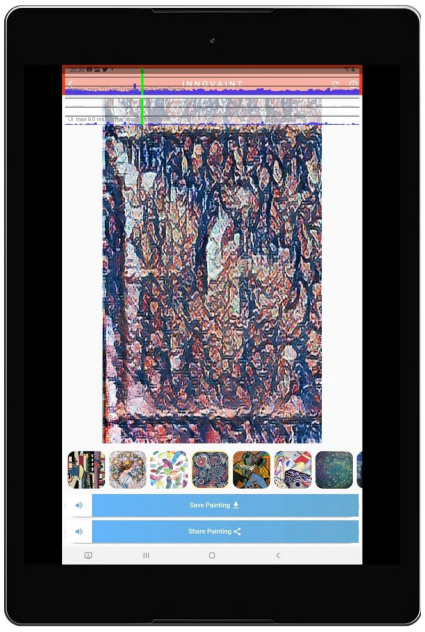


Figure F.4: Performance Overlay View - Tab S6 Lite

Figure F.5: Dart Dev Tools Performance View - Note 20 Ultra

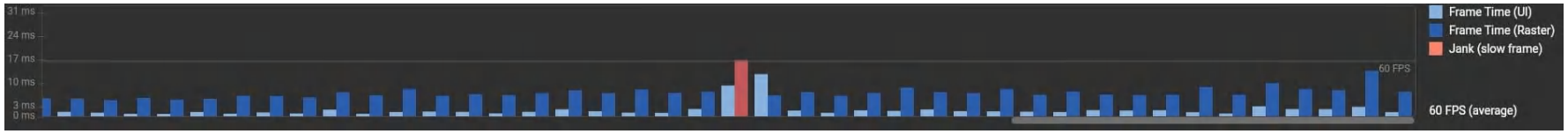


Figure F.6: Dart Dev Tools Performance View - S10 Plus

Figure F.7: Dart Dev Tools Performance View - Note 9

Figure F.8: Dart Dev Tools Performance View - Tab S6 Lite

Appendix G

Ethical Clearance Letter

Appendix H

Participant Information Sheet

Appendix I

User Guide

I.1 Introduction

Innovaint is a mobile Augmented Reality (AR) painting application for people with aphasia to aid communication therapy using art expression. The application allows users to create paintings by placing virtual objects in the real world. Users can curate various scenes as per desire, capture the augmented landscape and choose from a range of painting styles to apply to generate a painting.

I.2 Compatibility

The application is compatible with devices running the Android Operating System with versions higher than Android Pie (Android 9.0).

I.3 Application Demonstration Videos

The demo videos present a walk-through of a typical user session for a first time user. The videos can be found at the links below:

AR Interface

https://youtu.be/JlN963MMWTc

AST Interface

https://youtu.be/fuiBbftQ-yQ

The same videos can also be accessed from the help section within the application for ease.

I.4 Installation

The application could be installed through an Android Application Package (APK) file using a built-in package installer.

I.5 Application Overview

I.5.1 Home Screen

This is the home screen of the application.

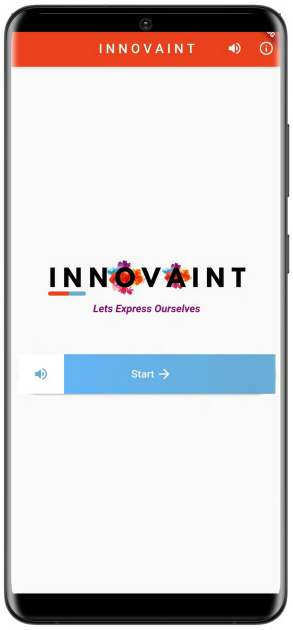


Figure I.1: Home Screen

Actions

Start Button Proceeds to the Augmented Reality Interface

Speaker Icon Reads aloud the displayed text on the screen

Help Icon Navigates to the help section

I.5.2 Augmented Reality Interface

This is the screen with the Augmented Reality Interface to curate scenes in the augmented environment.

Figure I.2: AR Interface

Actions

Delete Button Deletes the object from the scene

Capture Button Takes a snapshot of the scene

Place Button Toggles the objects' scroll view

Scale Button Toggles the scale slider

Rotate Button Toggles the rotate slider

Objects' Scroll View Displays the virtual objects

Scale Slider Changes the size of the object

Rotate Slider Changes the angle of the object

Next Button Proceeds to the Artistic Style Transfer Interface

Speaker Buttons Reads aloud the displayed text on the respective buttons

I.5.3 Content Image Screen

This is the screen where the user selects a content image for applying the style transfer to.

Actions

Confirm Image Button Selects the displayed image and proceeds to the style transfer screen

Change Image Button Displays the gallery to allow selection of a new image

Speaker Buttons Reads aloud the displayed text on the respective buttons

Help Icon Navigates to the help section

Back Icon Navigates to the last screen

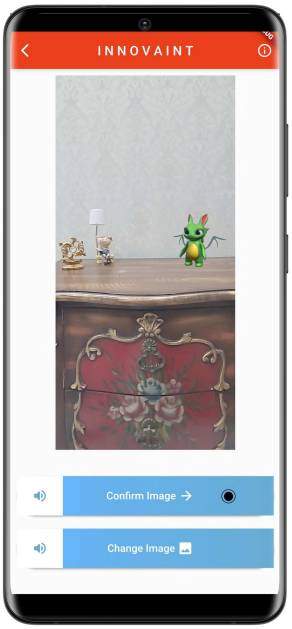


Figure I.3: Content Image Screen

I.5.4 Style Transfer Screen

This screen allows the user to generate paintings by applying the style transfer

Actions

Style Filters' List View Displays the painting styles

Save Painting Button Saves the generated painting to the gallery

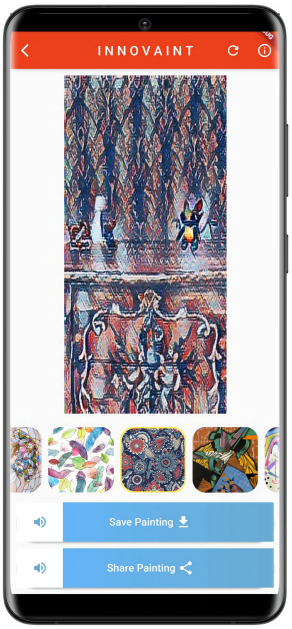


Figure I.4: Style Transfer Screen

Share Painting Button Shares the painting to external applications

Speaker Buttons Reads aloud the displayed text on the respective buttons

Revert Icon Reverts to the original content image

Help Icon Navigates to the help section

Back Icon Navigates to the last screen

I.5.5 Help Screen

Video Views Displays the demo videos

Play/Pause Button Plays or Pauses the Video

Mute Button Mutes or Unmutes the video

Full screen Button Enables full screen video view

Speaker Icon Reads aloud the displayed text on the screen

Home Icon Navigates to the home screen

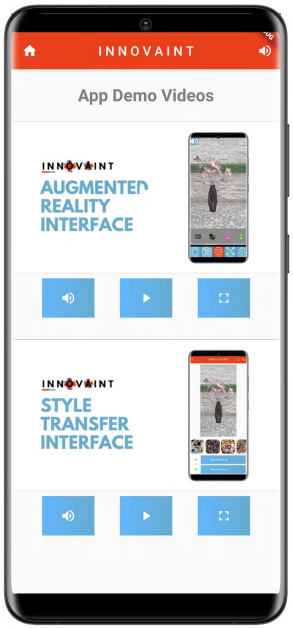


Figure I.5: Help Screen

I.6 Features

The application features are detailed in chapter 6 of this report.

Appendix J

Source Code

J.1 C# Scripts for AR Interface

J.1.1 Place and Drag Object

```
// Script to Place and Drag Objects
// Adapted from https://github.com/Shubhra22/FurnitureAR
// Adapted from https://learn.unity.com/tutorial/placing-and-manipulating-objects-in-ar

// Import Packages
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.EventSystems;
using UnityEngine.XR.ARFoundation;

public class InputHandler : MonoBehaviour
{
    // Define Variables
    [SerializeField] private Camera arCam;
    [SerializeField] private ARRaycastManager _raycastManager;
    [SerializeField] private GameObject crosshair;

    private List<ARRaycastHit> _hits = new List<ARRaycastHit>();

    private Touch touch;
    private Pose pose;
```

```
24      [SerializeField] GameObject spawnablePrefab;
25      GameObject spawnedObject;
27      // Start is called before the first frame update
28      void Start()
29      {
30          spawnedObject = null;
31          arCam = GameObject.Find("AR Camera").GetComponent<Camera>();
32      }
34      // Update is called once per frame
35     void Update()
36      {
37          // Recalculate Crosshair Position
38          CrosshairCalculation();
40          touch = Input.GetTouch(0);
42          // Ignore Touch over UI Element
43          if (IsPointerOverUI(touch)) return;
45          spawnablePrefab = DataManager.Instance.GetModel();
47          if (Input.touchCount < 0)
48              return;
50          RaycastHit hit;
51          Ray ray = arCam.ScreenPointToRay(Input.GetTouch(0).position);
53          // If a placed object is touched - Select Object
54          // If there is no placed object - Place New Object
55          // If finger is sliding - Drag Object
57          if (_raycastManager.Raycast(Input.GetTouch(0).position, _hits))
58          {
59              if (Input.GetTouch(0).phase == TouchPhase.Began && spawnedObject ==
    null)
60              {
61                  if (Physics.Raycast(ray, out hit))
62                  {
63                      if (hit.collider.gameObject.tag == "Spawnable")
```

```
64                  {
65                      spawnedObject = hit.collider.gameObject;
66                  }
67                  else
68                  {
69                      spawnedObject = Instantiate(DataManager.Instance.
    GetModel(), pose.position,
70                          DataManager.Instance.GetModel().transform.rotation);
71                  }
72              }
73          }
74          else if (Input.GetTouch(0).phase == TouchPhase.Moved &&
    spawnedObject != null)
75          {
76              spawnedObject.transform.position = _hits[0].pose.position;
77          }

79          if (Input.GetTouch(0).phase == TouchPhase.Ended)
80          {
81              spawnedObject = null;
82          }
83      }
84  }

86  // Check if Touch is over UI Element
87  bool IsPointerOverUI(Touch touch)
88  {
89      PointerEventData eventData = new PointerEventData(EventSystem.current);
90      eventData.position = new Vector2(touch.position.x, touch.position.y);
91      List<RaycastResult> results = new List<RaycastResult>();
92      EventSystem.current.RaycastAll(eventData, results);
93      return results.Count > 0;
94  }

96  // Update Crosshair Marker position
97  void CrosshairCalculation()
98  {
99      Vector3 origin = arCam.ViewportToScreenPoint(new Vector3(0.5f, 0.5f, 0))
    ;
100     Ray ray = arCam.ScreenPointToRay(touch.position);

102     if (_raycastManager.Raycast(ray, _hits))
```

```
        {
            pose = _hits[0].pose;
            crosshair.transform.position = pose.position;
            crosshair.transform.eulerAngles = new Vector3(90, 0, 0);
        }
    }
}
```

Listing J.1: C# Script for Placing and Dragging Object

J.1.2 Transformation Sliders

```
// Script to scale and rotate object using sliders
// Adapted from https://www.youtube.com/watch?v=43XRkmAmscM

// Import Packages
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class TransformSliders : MonoBehaviour
{
    // Define Variables
    public Slider scaleSlider;
    public Slider rotateSlider;
    public float scaleMinValue;
    public float scaleMaxValue;
    public float rotMinValue;
    public float rotMaxValue;

    public Transform Object;

    public static TransformSliders Instance;

    // Define Singleton
    private void Awake()
    {
        if (Instance == null)
            Instance = this;
        else if (Instance != this)
            Destroy(gameObject);
    }
```

```
    // Set Min Max Values and Slider Listener
    void Start()
    {
        scaleSlider.minValue = scaleMinValue;
        scaleSlider.maxValue = scaleMaxValue;
        scaleSlider.onValueChanged.AddListener(ScaleSliderUpdate);

        rotateSlider.minValue = rotMinValue;
        rotateSlider.maxValue = rotMaxValue;
        rotateSlider.onValueChanged.AddListener(RotateSliderUpdate);
    }

    // Update Object Size
    void ScaleSliderUpdate(float value)
    {
        Object.transform.localScale = new Vector3(value, value, value);
    }

    // Update Object Angle
    void RotateSliderUpdate(float value)
    {
        Object.transform.localEulerAngles = new Vector3(transform.rotation.x,
    value, transform.rotation.z);
    }
}
```

Listing J.2: C# Script for Transformation Sliders

J.1.3 Delete Object

```
// Script to Delete Placed Object

// Import Packages
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;

public class DestroyObject : MonoBehaviour
{
    // Define Variables
    public Button deleteBtn;
```

```
13      public GameObject Object;

15      public static DestroyObject Instance;

17      // Define Singleton
18      private void Awake()
19      {
20          if (Instance == null)
21              Instance = this;
22          else if (Instance != this)
23              Destroy(gameObject);
24      }

26      // Start is called before the first frame update
27      void Start()
28      {
29          if(deleteBtn!=null){
30              deleteBtn.onClick.AddListener(deleteObject);
31          }
32      }

34      // Delete Selected Object
35      private void deleteObject()
36      {
37          Destroy(Object);
38      }
39  }
```

Listing J.3: C# Script for Deleting Object

J.1.4 Capture Scene

```
1   // Script to Capture Augmented Scene

3   // Import packages
4   using System.Collections;
5   using System.Collections.Generic;
6   using UnityEngine;
7   using UnityEngine.UI;
8   using UnityEngine.XR.ARFoundation;

10  public class CaptureOnClick : MonoBehaviour
11  {
```

```
    // Define Variables
    private ARPlaneManager arPlaneManager;
    public Button cameraBtn;
    public GameObject canvas;
    public GameObject marker;

    public Text dt2;

    // Awake is run before the first frame
    void Awake()
    {
        arPlaneManager = GetComponent<ARPlaneManager>();
        if (cameraBtn != null)
        {
            cameraBtn.onClick.AddListener(CaptureScreen);
        }
    }

    // Disable UI Elements, take Snapshot and Save to Gallery, Inform Flutter
    private void CaptureScreen()
    {
        GetComponent<UnityMessageManager>().SendMessageToFlutter("capture");

        arPlaneManager.enabled = false;
        foreach (ARPlane plane in arPlaneManager.trackables)
        {
            plane.gameObject.SetActive(false);
        }

        ScreenCapture.CaptureScreenshot("arScene.png");

        canvas.gameObject.SetActive(false);
        marker.gameObject.SetActive(false);

        Invoke("EnablePlane", 0.3f);

        string existingPath = Application.persistentDataPath + "/arScene.png";

        NativeGallery.Permission permission = NativeGallery.SaveImageToGallery(
    existingPath, "unityApp", "arSc.png",
            (success, path) => dt2.text = "Media save result: " + success + " "
    + path);
```

```
52      }
54      // Enable UI Elements
55      void EnablePlane()
56      {
57          arPlaneManager.enabled = true;
58          foreach (ARPlane plane in arPlaneManager.trackables)
59          {
60              plane.gameObject.SetActive(true);
61          }
63          canvas.gameObject.SetActive(true);
64          marker.gameObject.SetActive(true);
65      }
66  }
```

Listing J.4: C# Script for Capturing Scene

J.2 Dart Code for AST Interface

J.2.1 AR Interface Screen

```
1  // Dart Code for AR Interface Screen
2  // Adapted from https://github.com/juicycleff/flutter-unity-view-widget
4  // Import Packages
5  import 'package:flutter/material.dart';
6  import 'package:flutter_tts/flutter_tts.dart';
7  import 'package:flutter_unity_widget/flutter_unity_widget.dart';
8  import 'package:flutter/services.dart';
10 class ARInterface extends StatefulWidget {
11   ARInterface({Key key}) : super(key: key);
13   @override
14   _ARInterfaceState createState() => _ARInterfaceState();
15 }
17 class _ARInterfaceState extends State<ARInterface> {
18   static final GlobalKey<ScaffoldState> _scaffoldKey =
19   GlobalKey<ScaffoldState>();
```

```
// Define UnityWidget Controller
UnityWidgetController _unityWidgetController;

@override
void initState() {
  SystemChrome.setEnabledSystemUIOverlays([]);
  super.initState();
}

@override
void dispose() {
  super.dispose();
  _unityWidgetController.dispose();
}

// Define Variables
bool pressed;
final ValueNotifier<int> camPressed = ValueNotifier<int>(0);
var pwdWidgets = <Widget>[];
final FlutterTts tts = FlutterTts();

@override
Widget build(BuildContext context) {
  SystemChrome.setEnabledSystemUIOverlays([]);

  // Set TTS Options
  tts.setLanguage('en');
  tts.setSpeechRate(0.6);
  tts.setVolume(1.0);

  pressed = true;

  return SafeArea(
    child: Scaffold(
      key: _scaffoldKey,
      body: Card(
          margin: const EdgeInsets.all(0),
          clipBehavior: Clip.antiAlias,
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(20.0),
          ),
          child: Stack(
```

```
63              children: [
64                // Unity Widget
65                UnityWidget(
66                  onUnityCreated: _onUnityCreated,
67                  onUnityMessage: onUnityMessage,
68                  onUnitySceneLoaded: onUnitySceneLoaded,
69                  fullscreen: true,
70                ),
71                // Next Button Listener
72                ValueListenableBuilder<int>(
73                  builder: (BuildContext context, int value, Widget child) {
74                    return Positioned(
75                      top: 0,
76                      right: 0,
77                      width: 55,
78                      child: value <= 0
79                          ? Container()
80                          : Container(
81                        height: 55.0,
82                        margin: EdgeInsets.all(0),
83                        child: RaisedButton(
84                          onPressed: () {
85                            _unityWidgetController.unload();
86                            Navigator.pushNamed(
87                                context, '/bloc_provider');
88                          },
89                          shape: RoundedRectangleBorder(
90                              borderRadius: BorderRadius.circular(0.0)),
91                          padding: EdgeInsets.all(0.0),
92                          child: Ink(
93                            decoration: BoxDecoration(
94                              gradient: LinearGradient(
95                                colors: [
96                                  Color(0xff64B6FF),
97                                  Color(0xff68ACD2)
98                                ],
99                                begin: Alignment.centerLeft,
100                               end: Alignment.centerRight,
101                             ),
102                           ),
103                           child: Container(
104                             constraints: BoxConstraints(
```

```
                                maxWidth: 55, minHeight: 50.0),
                            alignment: Alignment.center,
                            child: Row(
                              mainAxisAlignment:
                              MainAxisAlignment.center,
                              children: <Widget>[
                                Icon(
                                  Icons.arrow_forward,
                                  color: Colors.white,
                                )
                              ],
                            ),
                          ),
                        ),
                      ),
                    ),
                  );
                },
                valueListenable: camPressed,
              )
            ],
          )),
    ),
  );
}

// Unity Messages Handler
void onUnityMessage(message) {
  pressed = false;
  debugPrint('Received message from unity: ${message.toString()}');

  if (message.toString() == 'capture') {
    camPressed.value += 1;
  }

  if (message.toString() == 'CameraTts') {
    tts.speak('Capture Screen');
  }

  if (message.toString() == 'PlaceTts') {
    tts.speak('Place Objects');
  }
```

```
148     if (message.toString() == 'ScaleTts') {
149       tts.speak('Scale Object');
150     }

152     if (message.toString() == 'RotateTts') {
153       tts.speak('Rotate Object');
154     }

156     if (message.toString() == 'DeleteTts') {
157       tts.speak('Delete Object');
158     }
159   }

161   // unLoad Scene Handler
162   void onUnitySceneLoaded(SceneLoaded scene) {
163     print('Received scene loaded from unity: ${scene.name}');
164     print('Received scene loaded from unity buildIndex: ${scene.buildIndex}');
165   }

167   // Callback to connect created controller to the unity controller
168   void _onUnityCreated(controller) {
169     this._unityWidgetController = controller;
170   }
171 }
```

Listing J.5: Dart Code for AR Interface Screen

J.2.2 Image Selection Screen

```
1  // Dart Code for Image Selection Screen
2  // Adapted from https://github.com/PuzzleLeaf/flutter_tflite_style_transfer

4  // Import Packages
5  import 'dart:io';

7  import 'package:flutter/material.dart';
8  import 'package:flutter_bloc/flutter_bloc.dart';
9  import 'package:flutter_tts/flutter_tts.dart';
10 import '../blocs/image_bloc.dart';

12 import '../globals.dart';
```

```
class ImageScreen extends StatefulWidget {
  @override
  _ImageScreenState createState() => _ImageScreenState();
}

@override
void initState() {}

class _ImageScreenState extends State<ImageScreen> {
  final FlutterTts tts = FlutterTts();

  @override
  Widget build(BuildContext context) {

    // Set TTS Options
    tts.setLanguage('en');
    tts.setSpeechRate(0.6);
    tts.setVolume(1.0);

    // Define Back Navigation
    Future<bool> _onWillPop() async {
      imageCache.clear();
      var count = 0;
      Navigator.popUntil(context, (route) {
        return count++ == 2;
      });
    }

    return new WillPopScope(
      onWillPop: _onWillPop,
      child: new Scaffold(
        backgroundColor: Color(0xfffafafa),
        appBar: AppBar(
          backgroundColor: Color(0xffEC3F1B),
          centerTitle: true,
          title: Text('I N N O V A I N T'),
          leading: IconButton(
            onPressed: () {
              _onWillPop();
            },
            icon: Icon(
              Icons.arrow_back_ios,
```

```
            color: Colors.white,
          ),
        ),
        actions: <Widget>[
          IconButton(
              icon: Icon(Icons.info_outlined),
              onPressed: () {
                Navigator.pushNamed(context, '/help');
              }),
        ],
      ),
      body: SafeArea(
        child: Column(
          children: [
            SizedBox(height: 10.0),
            // Display Default Content Image
            Expanded(
              child: Image(
                image: FileImage(
                  File(
                      '/storage/emulated/0/Android/data/kcl.project.innovaint/
files/arScene.png'),
                ),
              ),
            ),
            SizedBox(height: 30),
            // Confirm Image Button Handler
            Container(
              height: 60.0,
              margin: EdgeInsets.all(10),
              child: RaisedButton(
                onPressed: () {
                  isClicked = true;
                  context.bloc<ImageBloc>().add(ImageEvent.loadImage());
                },
                shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(80.0)),
                padding: EdgeInsets.all(0.0),
                child: Ink(
                  decoration: BoxDecoration(
                    gradient: LinearGradient(
                      colors: [Color(0xff64B6FF), Color(0xff68ACD2)],
```

```
                        begin: Alignment.centerLeft,
                        end: Alignment.centerRight,
                      ),
                    ),
                    child: Container(
                      constraints: BoxConstraints(
                          maxWidth: double.infinity, minHeight: 50.0),
                      alignment: Alignment.center,
                      child: Row(
                        mainAxisAlignment: MainAxisAlignment.center,
                        children: <Widget>[
                          Container(
                            color: Colors.white,
                            height: double.infinity,
                            width: 70.0,
                            child: new IconButton(
                                icon: new Icon(
                                  Icons.volume_up_rounded,
                                  color: Color(0xff68ACD2),
                                ),
                                onPressed: () {
                                  tts.speak('Confirm Image');
                                }),
                          ),
                          Spacer(),
                          Text(
                            'Confirm Image',
                            style: TextStyle(
                              fontSize: 16,
                              color: Colors.white,
                            ),
                          ),
                          SizedBox(width: 3.0),
                          Icon(
                            Icons.arrow_forward,
                            color: Colors.white,
                          ),
                          Spacer(),
                          Container(
                            width: 55.0,
                          ),
                        ],
```

```
                    ),
                  ),
                ),
              ),
            ),
            // Change Image Button Handler
            Container(
              height: 60.0,
              margin: EdgeInsets.all(10),
              child: RaisedButton(
                onPressed: () {
                  isChangeClicked = true;
                  context.bloc<ImageBloc>().add(ImageEvent.loadImage());
                },
                shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(80.0)),
                padding: EdgeInsets.all(0.0),
                child: Ink(
                  decoration: BoxDecoration(
                    gradient: LinearGradient(
                      colors: [Color(0xff64B6FF), Color(0xff68ACD2)],
                      begin: Alignment.centerLeft,
                      end: Alignment.centerRight,
                    ),
                  ),
                  child: Container(
                    constraints: BoxConstraints(
                        maxWidth: double.infinity, minHeight: 50.0),
                    alignment: Alignment.center,
                    child: Row(
                      mainAxisAlignment: MainAxisAlignment.center,
                      children: <Widget>[
                        Container(
                          color: Colors.white,
                          height: double.infinity,
                          width: 70.0,
                          child: new IconButton(
                              icon: new Icon(
                                Icons.volume_up_rounded,
                                color: Color(0xff68ACD2),
                              ),
                              onPressed: () {
```

```
                                tts.speak('Change Image');
                              }),
                        ),
                        Spacer(),
                        Text(
                          'Change Image',
                          style: TextStyle(
                            fontSize: 16,
                            color: Colors.white,
                          ),
                        ),
                        SizedBox(width: 3.0),
                        Icon(
                          Icons.image,
                          color: Colors.white,
                        ),
                        Spacer(),
                        Container(
                          width: 55.0,
                        ),
                      ],
                    ),
                  ),
                ),
              ),
              SizedBox(height: 30),
            ],
          ),
        ),
      ),
    );
  }
}
```

Listing J.6: Dart Code for Image Selection Screen

J.2.3 Style Transfer Screen

```
// Dart Code for Style Transfer Screen
// Adapted from https://github.com/PuzzleLeaf/flutter_tflite_style_transfer

// Import Packages
```

```dart
import 'dart:io';
import 'dart:typed_data';

import 'package:flutter/material.dart';
import 'package:flutter_tts/flutter_tts.dart';
import 'package:flutter_bloc/flutter_bloc.dart';
import 'package:fluttertoast/fluttertoast.dart';
import 'package:image_gallery_saver/image_gallery_saver.dart';
import 'package:path_provider/path_provider.dart';
import 'package:share_plus/share_plus.dart';
import '../blocs/image_bloc.dart';

import '../globals.dart';

class TransferScreen extends StatefulWidget {
  @override
  _TransferScreenState createState() => _TransferScreenState();
}

class _TransferScreenState extends State<TransferScreen> {
  // Define Variables
  int selectStyle = -1;
  var imageBloc;
  final FlutterTts tts = FlutterTts();

  @override
  void initState() {
    imageBloc = context.bloc<ImageBloc>();
    if (true) {}
    super.initState();
  }

  @override
  Widget build(BuildContext context) {
    // Define Back Navigation
    Future<bool> _onWillPop() async {
      imageBloc.add(ImageEvent.resetImage());
    }

    // Set TTS Options
    tts.setLanguage('en');
    tts.setSpeechRate(0.6);
```

```
47    tts.setVolume(1.0);
48
49    return new WillPopScope(
50      onWillPop: _onWillPop,
51      child: new Scaffold(
52        backgroundColor: Color(0xfffafafa),
53        appBar: AppBar(
54          backgroundColor: Color(0xffEC3F1B),
55          centerTitle: true,
56          title: Text('I N N O V A I N T'),
57          leading: IconButton(
58            onPressed: () {
59              imageBloc.add(ImageEvent.resetImage());
60            },
61            icon: Icon(
62              Icons.arrow_back_ios,
63              color: Colors.white,
64            ),
65          ),
66          actions: <Widget>[
67            IconButton(
68                icon: Icon(Icons.refresh_rounded),
69                onPressed: () {
70                  isOriginalClicked = true;
71                  imageBloc.add(ImageEvent.resetImage());
72                }),
73            IconButton(
74                icon: Icon(Icons.info_outlined),
75                onPressed: () {
76                  Navigator.pushNamed(context, '/help');
77                }),
78          ],
79        ),
80        body: SafeArea(
81          child: BlocBuilder<ImageBloc, ImageState>(
82            builder: (context, state) {
83              return Stack(
84                children: [
85                  Column(
86                    children: [
87                      SizedBox(height: 10.0),
88                      // Display Painting
```

```
                    Expanded(
                      child: state.transferImage == null
                          ? Image.memory(state.originImage)
                          : Image.memory(state.transferImage),
                    ),
                    SizedBox(
                      height: 20,
                    ),
                    // Display Painting Styles
                    Container(
                      height: 100,
                      child: ListView.builder(
                        padding: const EdgeInsets.symmetric(horizontal: 10),
                        scrollDirection: Axis.horizontal,
                        itemBuilder: (context, index) {
                          var stylePath = 'assets/images/style$index.jpg';
                          return GestureDetector(
                            onTap: () {
                              if (selectStyle == index) {
                                return;
                              }
                              setState(() {
                                selectStyle = index;
                              });
                              imageBloc
                                  .add(ImageEvent.transferImage(stylePath));
                            },
                            child: Container(
                              margin:
                                  const EdgeInsets.symmetric(horizontal: 5),
                              decoration: BoxDecoration(
                                border: selectStyle == index
                                    ? Border.all(
                                        color: Colors.yellow,
                                        width: 2,
                                      )
                                    : null,
                                borderRadius: BorderRadius.circular(20),
                              ),
                              child: ClipRRect(
                                borderRadius: BorderRadius.circular(20),
                                child: Image.asset(stylePath),
```

```
                                ),
                              ),
                            );
                          },
                          itemCount: 20,
                        ),
                      ),
                      // Save Painting Button Handler
                      Container(
                        height: 60.0,
                        margin: EdgeInsets.fromLTRB(10.0, 15.0, 10.0, 0.0),
                        child: RaisedButton(
                          onPressed: () async {
                            if (state.transferImage != null) {
                              await saveImage(state.transferImage);
                              tts.speak('Image Saved');
                              Fluttertoast.showToast(
                                  msg: "Image Saved to Gallery",
                                  toastLength: Toast.LENGTH_SHORT,
                                  gravity: ToastGravity.BOTTOM,
                                  timeInSecForIosWeb: 1,
                                  backgroundColor: Color(0xffEC3F1B),
                                  textColor: Colors.white,
                                  fontSize: 16.0);
                            } else {
                              tts.speak('Please Apply Style');
                              Fluttertoast.showToast(
                                  msg: "Please Appyl a Style First",
                                  toastLength: Toast.LENGTH_SHORT,
                                  gravity: ToastGravity.BOTTOM,
                                  timeInSecForIosWeb: 1,
                                  backgroundColor: Colors.red,
                                  textColor: Colors.white,
                                  fontSize: 16.0);
                            }
                          },
                          shape: RoundedRectangleBorder(
                              borderRadius: BorderRadius.circular(80.0)),
                          padding: EdgeInsets.all(0.0),
                          child: Ink(
                            decoration: BoxDecoration(
                              gradient: LinearGradient(
```

```
                colors: [Color(0xff64B6FF), Color(0xff68ACD2)],
                begin: Alignment.centerLeft,
                end: Alignment.centerRight,
              ),
            ),
            child: Container(
              constraints: BoxConstraints(
                  maxWidth: double.infinity, minHeight: 50.0),
              alignment: Alignment.center,
              child: Row(
                mainAxisAlignment: MainAxisAlignment.center,
                children: <Widget>[
                  Container(
                    color: Colors.white,
                    height: double.infinity,
                    width: 70.0,
                    child: new IconButton(
                        icon: new Icon(
                          Icons.volume_up_rounded,
                          color: Color(0xff68ACD2),
                        ),
                        onPressed: () {
                          tts.speak('Save Painting');
                        }),
                  ),
                  Spacer(),
                  Text(
                    'Save Painting',
                    style: TextStyle(
                      fontSize: 16,
                      color: Colors.white,
                    ),
                  ),
                  SizedBox(width: 3.0),
                  Icon(
                    Icons.file_download,
                    color: Colors.white,
                  ),
                  Spacer(),
                  Container(
                    width: 55.0,
                  ),
```

```
                                ],
                              ),
                            ),
                          ),
                        ),
                      ),
                      // Share Painting Button Handler
                      Container(
                        height: 60.0,
                        margin: EdgeInsets.all(10),
                        child: RaisedButton(
                          onPressed: () async {
                            if (state.transferImage != null) {
                              await saveAndShare(state.transferImage);
                            } else {
                              await saveAndShare(state.originImage);
                            }
                          },
                          shape: RoundedRectangleBorder(
                              borderRadius: BorderRadius.circular(80.0)),
                          padding: EdgeInsets.all(0.0),
                          child: Ink(
                            decoration: BoxDecoration(
                              gradient: LinearGradient(
                                colors: [Color(0xff64B6FF), Color(0xff68ACD2)],
                                begin: Alignment.centerLeft,
                                end: Alignment.centerRight,
                              ),
                            ),
                            child: Container(
                              constraints: BoxConstraints(
                                  maxWidth: double.infinity, minHeight: 50.0),
                              alignment: Alignment.center,
                              child: Row(
                                mainAxisAlignment: MainAxisAlignment.center,
                                children: <Widget>[
                                  Container(
                                    color: Colors.white,
                                    height: double.infinity,
                                    width: 70.0,
                                    child: new IconButton(
                                        icon: new Icon(
```

```
                                    Icons.volume_up_rounded,
                                    color: Color(0xff68ACD2),
                                  ),
                                  onPressed: () {
                                    tts.speak('Share Painting');
                                  }),
                            ),
                            Spacer(),
                            Text(
                              'Share Painting',
                              style: TextStyle(
                                fontSize: 16,
                                color: Colors.white,
                              ),
                            ),
                            SizedBox(width: 3.0),
                            Icon(
                              Icons.share,
                              color: Colors.white,
                            ),
                            Spacer(),
                            Container(
                              width: 55.0,
                            )
                          ],
                        ),
                      ),
                    ),
                  ),
                ),
              ],
            ),
            Center(
              child: state.isLoading ? _loadingWidget() : Container(),
            )
          ],
        );
      },
    ),
  ),
);
```

```
  }

  // Display Processing Container
  Widget _loadingWidget() {
    return FittedBox(
      child: Container(
        padding: const EdgeInsets.symmetric(horizontal: 10, vertical: 10),
        alignment: Alignment.center,
        decoration: BoxDecoration(
          color: Colors.black.withOpacity(0.6),
          borderRadius: BorderRadius.circular(10),
        ),
        child: Text(
          'Processing...',
          style: TextStyle(
            color: Colors.white,
            fontSize: 20,
          ),
        ),
      ),
    );
  }

  // Save Painting to Gallery
  Future<String> saveImage(Uint8List bytes) async {
    final result = await ImageGallerySaver.saveImage(bytes);
    return result['filePath'];
  }

  // Share Painting Externally
  Future saveAndShare(Uint8List bytes) async {
    final directory = await getApplicationDocumentsDirectory();
    final image = File('${directory.path}/styledImage.png');
    image.writeAsBytes(bytes);

    await Share.shareFiles([image.path]);
  }
}
```

Listing J.7: Dart Code for Style Transfer Screen