A PROJECT REPORT

On

“ FloatArt: LET YOUR IMAGINATION TAKE A VIRTUAL FLIGHT! “

Submitted to

KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

BACHELOR’S DEGREE IN
COMPUTER SCIENCE & COMMUNICATION ENGINEERING

NAME	ROLL
1. AKASH CHANDRAKAR	2129016
2. RISHABH MOHATA	2129032
3. TUSHAR BHATT	2129119

BY

UNDER THE GUIDANCE OF
Dr. Sushruta Mishra
(Assistant Professor)

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAR, ODISHA - 751024
November 2024

KIIT Deemed to be University
School of Computer Engineering
Bhubaneswar, ODISHA 751024

CERTIFICATE

This is to certify that the project entitled

“ FloatArt: LET YOUR IMAGINATION TAKE A VIRTUAL FLIGHT! ”

submitted by

NAME	**ROLL**
1. AKASH CHANDRAKAR	2129016
2. RISHABH MOHATA	2129032
3. TUSHAR BHATT	2129119

is a record of Bonafide work carried out by them, in the partial fulfilment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science & Communication Engineering) at KIIT Deemed to be university, Bhubaneswar. This work is done during the year 2024-2025, under our guidance.

Date: 25/11/24

Dr. Sushruta Mishra
(Assistant Professor)
Project Guide

ACKNOWLEDGEMENT

We extend our deepest gratitude to **Dr. Sushruta Mishra** for his expert guidance, valuable insights, and continuous encouragement throughout this project. His unwavering support and dedication have been instrumental in shaping our work from its inception to its successful completion.

We deeply appreciate his patience and constructive feedback, which inspired us to strive for excellence at every stage. This accomplishment would not have been possible without his belief in our abilities and willingness to guide us.

We are also thankful to everyone who contributed indirectly, offering support and motivation.

FROM

1. AKASH CHANDRAKAR	2129016
2. RISHABH MOHATA	2129032
3. TUSHAR BHATT	2129119

ABSTRACT

FloatArt is a trailblazing virtual jamboard designed to revolutionize digital creative environments by leveraging the power of machine learning (ML), deep learning (DL), and artificial intelligence (AI). This innovative platform offers a touchless interface that enables users to engage in intuitive drawing, writing, and collaborative tasks through advanced hand gesture recognition technology.

The system boasts a wide range of features, including infinite color palettes, precise shape correction, and customizable brush sizes, catering to diverse creative needs. It integrates speech-to-text recognition, enabling seamless transcription of spoken words into written text, and incorporates OpenAI-powered internet search to provide real-time knowledge and insights.

By analyzing user inputs in real time, FloatArt refines shapes and gestures, ensuring unparalleled precision and usability in digital workspaces. This platform is designed to enhance creative and collaborative efforts, making it particularly valuable for artists, designers, and educators.

FloatArt redefines virtual collaboration by providing a versatile, high-tech solution that empowers users to explore their creativity while boosting productivity and efficiency in digital environments.

KEYWORDS: FloatArt, Virtual Jamboard, Machine Learning, Deep Learning, Artificial Intelligence, Touchless Interaction, Hand Gesture Recognition, Shape Correction, Speech-to-Text, Collaborative Workspaces.

CONTENTS

Chapter 1

INTRODUCTION

The digital world is always changing, stretching the limits of cooperation and ingenuity. The spectrum of artistic expression is often limited by traditional tools, which necessitate repetitive manual inputs or physical engagement. Presenting FloatArt, a the moment virtual jamboard that revolutionizes digital creative by providing users with a straightforward to operate, touchless platform. Fundamentally, FloatArt harnesses novel technologies such as Artificial Intelligence (AI), Deep Learning (DL), and Machine Learning (ML) to provide a smooth drawing, writing, and teamwork experience. Users can interact with the platform using hand gestures thanks to the system's touchless interface, which makes idea presentation organic and futuristic.

By mixing capabilities like speech-to-text transcription, accurate shape correction, unconstrained color palettes, and OpenAI-powered search, FloatArt surpasses standard functionality. Without the limitations of conventional tools, these attributes offer professionals, educators, and creators an excellent setting in which to realize their ideas. FloatArt is an essential tool for nowadays because it combines accuracy, accessibility, and imaginative thinking to improve digital creative settings and revolutionize virtual collaboration. It seeks to close the gap between invention and delivery, encouraging cooperation and originality in a variety of fields.

Benefits of virtual jamboards	Description
Enhanced Collaboration	Allows multiple users to work together in real-time from different locations, fostering teamwork.
Touchless Interaction	Facilitates contactless drawing and writing, promoting hygiene and accessibility.
Cost-Effectiveness	Reduces the need for physical materials like whiteboards, markers, or papers.
Eco-Friendliness	Minimizes paper waste by providing a sustainable digital alternative.
Customization	Offers advanced features like color palettes, shapes, and text options for personalized creativity.
Cross-Platform Access	Can be used on various devices, making it convenient for remote learning or work.
Real-Time Feedback	Instant updates and shared visuals enable faster decision-making and brainstorming.
Engaging Learning Experiences	Enhances virtual learning with interactive features, making lessons more engaging and interactive for students.

Chapter 2
BASIC CONCEPTS

2.1 Goal

The FloatArt project aims to develop a user-friendly, touchless virtual jamboard that uses advanced technologies like artificial intelligence (AI), deep learning (DL), and machine learning (ML) to foster creativity and teamwork. Through groundbreaking features like gesture detection, configurable tools, speech-to-text integration, and internet-powered search, it seeks to offer a smooth environment for writing, drawing, and brainstorming. The initiative aims at transforming digital workspaces by enhancing accuracy, usability, and accessibility, encouraging creativity and productivity for designers, artists, educators, and collaborators.

2.2 Need of the application

1. **Limited Interaction Modes**: Traditional tools require physical input through styluses, mice, or keyboards, which can be restrictive and cumbersome.
2. **Precision Issues**: Many platforms lack features like shape correction or intuitive gesture recognition, leading to inaccurate outputs.
3. **Absence of Customization**: Limited color options, fixed brush sizes, and lack of advanced features restrict creative freedom.
4. **Inefficient Collaboration**: Real-time collaboration tools are often laggy or lack essential features for seamless teamwork.
5. **Inconvenient Text Integration**: Existing platforms do not provide efficient speech-to-text capabilities for annotating or adding notes.
6. **Difficulty in Accessing External Resources**: Switching between applications to search for information disrupts workflow.

Solutions Offered by FloatArt

1. **Touchless Interaction**: Gesture-based drawing and interaction eliminate the need for physical devices, offering a more natural and accessible experience.
2. **Enhanced Precision**: Advanced shape correction and machine learning algorithms ensure accurate and refined outputs.
3. **Rich Customization**: Features like infinite color palettes, adjustable brush sizes, and multiple drawing tools enhance creative possibilities.
4. **Seamless Collaboration**: Real-time updates and shared access allow multiple users to work simultaneously without interruptions.
5. **Speech-to-Text Integration**: Effortlessly converts spoken words into text, simplifying the addition of annotations or notes.
6. **Integrated Internet Search**: AI-powered search enables users to find information or references directly within the platform.

By solving these issues, FloatArt provides a robust and innovative solution for digital creativity and collaboration, meeting the needs of artists, designers, educators, and remote teams.

2.3 Scope

Application	Description
Artists and Designers	Provides a touchless interface for sketching, creating designs, and refining digital artwork with precision tools like shape correction and customizable brushes.
Educators and Students	Facilitates interactive and engaging virtual classrooms with features like speech-to-text for notes, real-time collaboration, and easy sharing of ideas on a virtual board.
Collaborative Workspaces	Enhances brainstorming and team discussions in remote or hybrid work environments by enabling multiple users to interact on the same canvas in real-time.
Accessibility	Promotes inclusivity by offering touchless interaction, making it ideal for users with physical disabilities or those who prefer hygienic, contact-free tools.
Advanced Technology Integration	Integrates AI-powered search, machine learning-driven gesture recognition, and speech-to-text to create an intuitive and futuristic workspace.
Sustainability	Encourages eco-friendly practices by replacing physical tools like whiteboards, markers, and paper, reducing material waste.
Global Reach	Supports multi-language capabilities, making it suitable for diverse audiences and global users in educational institutions, creative industries, and professional environments.
Future Development	Provides a foundation for incorporating advanced features like AR/VR integration, cloud-based collaboration, and enhanced AI capabilities for predictive design assistance.

Chapter 3
LITERATURE SURVEY

Key developments in air-writing systems and virtual drawing tools have shaped the foundation for FloatArt's design and technology. These studies provide insights into touchless interaction, gesture recognition, and their applications in creative environments.

In [1], an air-writing system for smart glasses utilized a region-based convolutional neural network (R-CNN) with an optimized MobileNetV2 to achieve real-time fingertip localization and efficient gesture recognition. Similarly, [2] introduced "Air Doodle," a tool that eliminates the need for physical notebooks by enabling real-time digital drawing and content sharing, enhancing collaborative workflows. The method in [3] leveraged a video-based approach for air-writing using mobile cameras to track colored fingertips and apply Optical Character Recognition (OCR), promoting a natural, device-free interaction. Additionally, [4] proposed tracking a red LED mounted on a user’s finger with a web camera to draw and identify characters, relying on precise environmental conditions for accuracy. [5] presented an augmented desk interface combining a video projector and CCD camera to enable fingertip-based desktop application control, with predefined search windows to improve fingertip detection efficiency.[6] The Leap Motion controller enables precise hand and finger tracking but lacks built-in functionality for recognizing air-drawn letters. A proposed solution uses Deep Belief Networks (DBN) with Resilient Backpropagation (Rprop) fine-tuning, achieving 99.71% accuracy on a dataset of 30,000 handwritten letters. Rprop significantly improves both speed and precision compared to standard Backpropagation, making it an effective method for air-writing recognition.

These studies highlight the progress in gesture-based systems and the potential of touchless technologies to enhance digital collaboration and creativity, forming the backbone of FloatArt’s vision.

Chapter 4
PLATFORM SPECIFICATIONS

Software:

Category	Specifications
Operating System	Windows (7/8/8.1/10/11), Linux distributions
User Interface	HTML5, CSS3, Bootstrap v5.3 for enhanced responsiveness
Client-side Scripting	JavaScript
Programming Language	Python (for backend and ML/DL integration)
Coding Platform	Visual Studio Code (VS Code)
Execution Method	Direct execution of Python code launches the canvas interface
Web Browser	Google Chrome, Microsoft Edge, Mozilla Firefox (for testing auxiliary web features)
APIs and Libraries	MediaPipe for gesture recognition, OpenCV for image processing, TensorFlow/PyTorch for ML/DL
Additional Tools	SpeechRecognition library for voice input, OpenAI API for speech-to-text and internet search
Features Supported	Infinite color palettes, shape correction, customizable brush sizes, gesture-based input, collaboration
Connection Link	Standalone Python script execution launches FloatArt canvas locally; no external connection is required.

Hardware:

Memory	6 GB
Processor Type	Intel Pentium, i3, i5, i7 or faster
Processor Speed	1.83 GHz or faster processor, Intel Pentium compatible
Swap Space	2.3 GB
Hard Disk Space	500GB/or less
Display	16 Bit Color

Chapter 5
PROBLEM STATEMENT

The current system for virtual drawing has significant limitations that hinder its flexibility and usability. It primarily relies on finger-based interactions, making it incompatible with tools like crayons, paints, or styluses. This restriction arises from the challenge of isolating fingers from RGB images without depth sensors, which also prevents tracking the pen's position relative to the surface or capturing upward and bottom-up movements. Relying on a single RGB camera adds further limitations, as it lacks precision in diverse environments.

Additionally, the system generates abstract, model-unseen images due to inaccuracies in tracking finger paths. Transitioning the drawing process between regions in real-time requires extensive coding effort and computational resources. Users must also learn complex gestures for effective control, making it less intuitive, especially for non-technical individuals. Other challenges include reduced precision, performance degradation in poor lighting conditions, lack of multi-user functionality for collaboration, and ergonomic issues during prolonged use. These constraints highlight the need for a more adaptive and user-friendly solution that enhances real-time gesture recognition and usability while addressing environmental and computational challenges.

Chapter 6
REQUIREMENTS SPECIFICATIONS

6.1 Requirements

6.1.1 Functional Requirements

a) ***Gesture-Based Drawing:*** Enable users to create drawings, shapes, and text on the canvas using hand gestures, offering a touchless interaction mode.
b) ***Finger Path Detection:*** Accurately identify finger movements and paths for precise and smooth drawing in real-time.
c) ***Dynamic Shape Recognition:*** Automatically recognize and refine basic shapes, such as circles and squares, drawn by the user.
d) ***Tool Options*:** Include a variety of tools, such as pens, brushes, erasers, and text insertion features.
e) ***Undo/Redo Functionality:*** Provide the ability to undo and redo actions to allow for mistake correction and better control over the creation process.

f) ***Color Palette Integration:*** Offer an extensive range of colors for the user to choose from, ensuring creative flexibility.
g) ***Canvas Customization:*** Allow users to modify the canvas background and save their creations in multiple formats.
h) ***Voice Command Support:*** Implement speech-to-text functionality for text input and commands to enhance usability.
i) ***Save and Export Options:*** Enable users to save and export their work in formats like PNG, JPEG, or PDF for sharing or printing.
j) ***Real-Time Feedback:*** Provide immediate visual feedback for every gesture or action performed by the user on the canvas.

6.1.2 Non-Functional Requirements

- ✧ ***Performance:*** Ensure the application processes gestures and finger movements in real-time without noticeable delays, providing a seamless user experience.
- ✧ ***Scalability:*** Support additional features, tools, or integrations (e.g., AI enhancements or cloud storage) as the application evolves.
- ✧ ***Portability:*** The system should function on multiple platforms, including Windows, macOS, and Linux, without requiring significant modifications.
- ✧ ***Usability:*** The interface should be intuitive, requiring minimal effort for users to learn and operate.
- ✧ ***Reliability:*** Ensure the application runs without crashes or errors, even under extended usage or heavy drawing loads.
- ✧ ***Security:*** Protect the system from unauthorized access or modifications, ensuring user-created content remains safe.
- ✧ ***Compatibility:*** Operate effectively with standard web cameras and hardware, without requiring specialized equipment.
- ✧ ***Maintainability:*** Facilitate easy updates, bug fixes, and system improvements through clean and modular code.
- ✧ ***Energy Efficiency*:** Optimize the application to consume minimal system resources, ensuring smooth operation on devices with limited hardware capacity.
- ✧ ***Accessibility:*** Provide features like customizable contrast settings and voice command capabilities to accommodate users with diverse needs.

6.2 Constraints

- ✓ The application only supports standard RGB cameras without requiring depth sensors, limiting advanced tracking capabilities.
- ✓ Real-time gesture recognition accuracy may decrease under poor lighting or cluttered backgrounds.
- ✓ Works effectively within the camera's field of view; gestures beyond this range cannot be processed.
- ✓ The application may not be compatible with all devices due to variations in hardware and operating systems.
- ✓ Continuous hand tracking may require optimal system resources to maintain smooth performance.

6.3 Assumptions

- ❖ Users are equipped with a functional webcam or camera compatible with the system.
- ❖ The application is primarily used in environments with stable lighting conditions for reliable performance.
- ❖ Users have a basic understanding of gesture-based interactions.
- ❖ Internet connectivity is available for features like speech-to-text and collaborative functionality.
- ❖ The system will be used for creative, educational, or professional purposes within its intended scope.

6.4 Dependencies

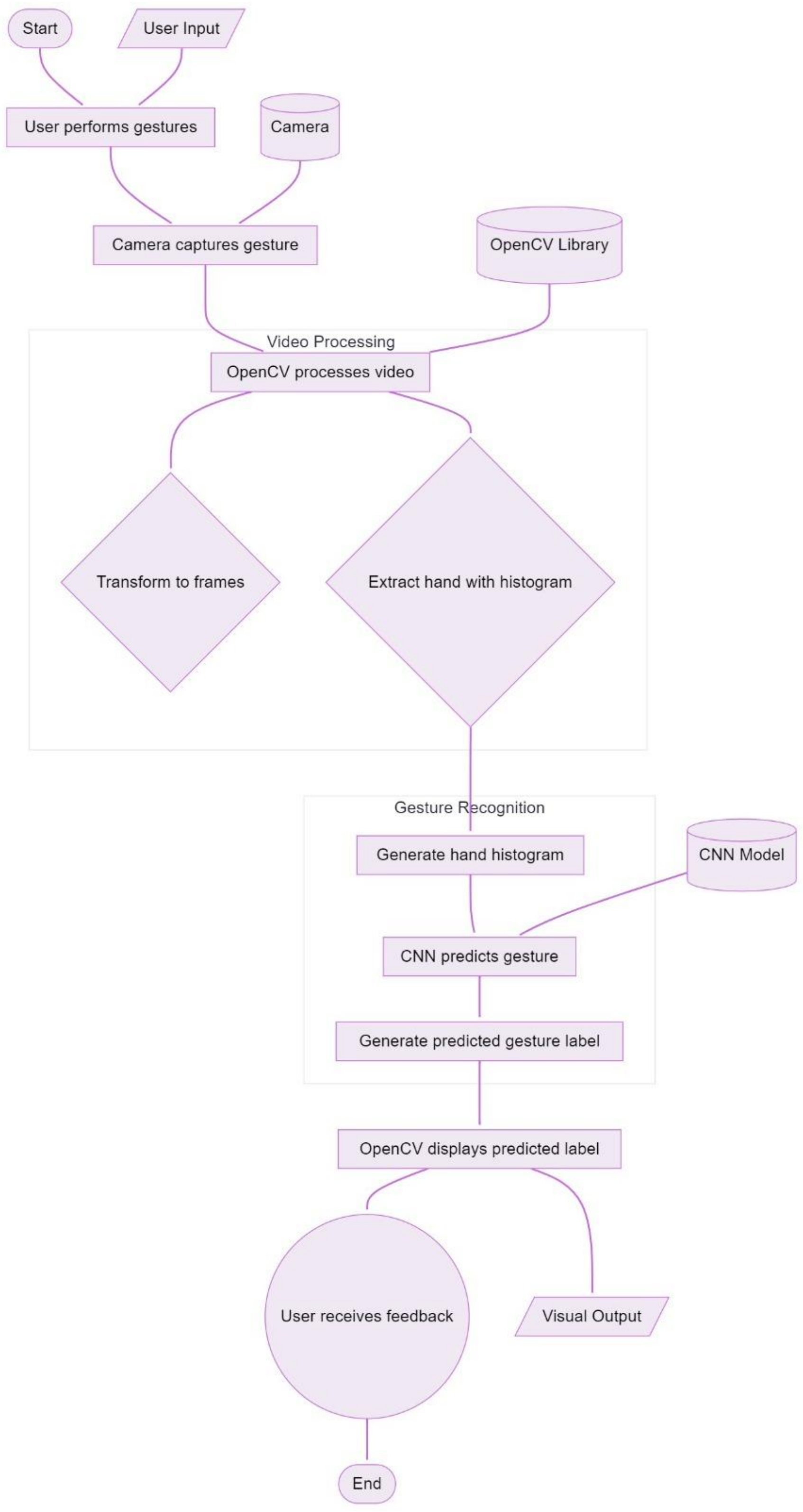
Dependency Type	Details
Hardware	✓ RGB camera or webcam for gesture recognition. ✓ Device capable of running real-time computer vision models.
Software	✓ Compatible OS (Windows/Linux) ✓ Python libraries: * MediaPipe (for gesture recognition) * OpenCV (for image processing) * NumPy (for numerical computations) * PyAudio (for voice input) ✓ Internet connection (for collaborative features and online resources)
Environment	✓ Proper lighting for clear detection. ✓ Minimal background clutter to avoid recognition noise.
User	✓ Basic understanding of gesture-based systems and drawing tools.
System	✓ Stable camera frame rates and calibration for accurate tracking.

6.5 Calibration

- Calibration and accuracy are critical in interactive systems, especially those involving digital canvases or gesture recognition. Calibration ensures that input devices, such as touchscreens or motion sensors, align with on-screen elements, reducing discrepancies between intended and executed actions.
- High accuracy is essential for precise drawing, painting, or gesture-based interactions. To maintain performance and user satisfaction, regular calibration checks and fine-tuning mechanisms are necessary, ensuring that user actions are faithfully translated into digital outputs. This improves the system's usability and the overall user experience.

Chapter 7

PROPOSED MODEL



1. Start and User Input:

- **User Input**: The system starts by waiting for the user to perform a gesture. This can be done using the webcam or camera feed.
- The user interacts by performing a hand gesture that will be captured by the camera.

2. Camera Captures Gesture:

- **Camera**: The camera is set up to capture the user's hand gesture in real-time. In this step, the video feed from the camera is processed.
- You would use OpenCV's `cv2.VideoCapture()` function to access the webcam feed and get each frame for processing.

```
import cv2

cap = cv2.VideoCapture(0)  # '0' for default camera
while True:
    ret, frame = cap.read()  # Capture each frame
    if not ret:
        break
    cv2.imshow("Video Feed", frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
```

3. Video Processing:

- **OpenCV Processes Video**: Once the video feed is captured, OpenCV processes the video stream to detect the hand and gestures.
- **Transform to Frames**: The captured video feed is divided into frames, which are processed individually. Each frame represents a snapshot of the real-time video.

```
ret, frame = cap.read()  # Read a frame
gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)  # Convert the frame to grayscale
```

4. Extract Hand with Histogram:

- **Hand Extraction**: To detect the hand from the background, a histogram-based approach is used (like the **Histogram of Oriented Gradients (HOG)** or **Hand Detection using skin color segmentation**). It can also include background subtraction to isolate the hand from the rest of the scene.

Hand Detection Code Example (using skin color):

```
# Convert frame to HSV color space for better segmentation of skin tone
hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
```

```python
# Define skin color range
lower_skin = np.array([0, 20, 70], dtype=np.uint8)
upper_skin = np.array([20, 255, 255], dtype=np.uint8)
# Mask for detecting skin
mask = cv2.inRange(hsv, lower_skin, upper_skin)
hand = cv2.bitwise_and(frame, frame, mask=mask)
```

- **Histogram Calculation**: The hand can also be analyzed by creating a **histogram** (e.g., using the color distribution or edge patterns).

```python
# Compute the histogram of the hand in HSV color space
hist = cv2.calcHist([hand], [0], None, [256], [0, 256])
```

5. Gesture Recognition:

- **Generate Hand Histogram**: The histogram generated in the previous step is used as a feature for gesture recognition.
- **CNN Predicts Gesture**: The histogram or hand features are fed into a **Convolutional Neural Network (CNN)** to predict the gesture. The CNN is trained to recognize various gestures (e.g., fist, open hand, peace sign, etc.).

```python
import tensorflow as tf
from tensorflow.keras.models import load_model
model = load_model('gesture_model.h5')
hand_resized = cv2.resize(hand, (64, 64))  # Resize to match the model's input
hand_normalized = hand_resized / 255.0  # Normalize the pixel values
hand_input = np.expand_dims(hand_normalized, axis=0)  # Add batch dimension
predicted_label = model.predict(hand_input)    # Predict the gesture
gesture = np.argmax(predicted_label, axis=1)  # Get the predicted gesture label
```

6. Generate Predicted Gesture Label:

- After the CNN makes its prediction, a label (or class) corresponding to the recognized gesture is generated. For example, if the CNN predicts a "thumbs up," the label might be `1`.

```python
gesture_labels = ['thumbs_up', 'peace', 'fist', 'open_hand']
predicted_gesture = gesture_labels[gesture[0]]
```

7. OpenCV Displays Predicted Label:

- **Visual Output**: The predicted label (gesture name) is displayed on the screen as feedback to the user.

```
cv2.putText(frame, predicted_gesture, (50, 50), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 255,
0), 2)
cv2.imshow("Gesture Recognition", frame)
```

Chapter 8
SYSTEM DESIGN

8.1 Design Approach

- ✓ System design outlines the methodical approach to building a new system. It comprises several steps that clarify the functional requirements and procedural details for the system's implementation as recommended in the feasibility study. The focus is on converting performance requirements into actionable design specifications while considering both logical and physical stages of development.
- ✓ The primary challenge is capturing an image from the external environment and processing it to extract the desired color essential for the system's functionality. To achieve this, the design must address dynamic motion tracking, which reflects real-time changes. The choice between static and dynamic gesture processing depends on client needs and situational requirements. Given the colorful nature of the captured image, it is necessary to separate color bands using Python's OpenCV library. Additionally, extracting color bands based on finger gestures with colored tapes requires model training. The system must operate in real time to accurately identify finger gestures

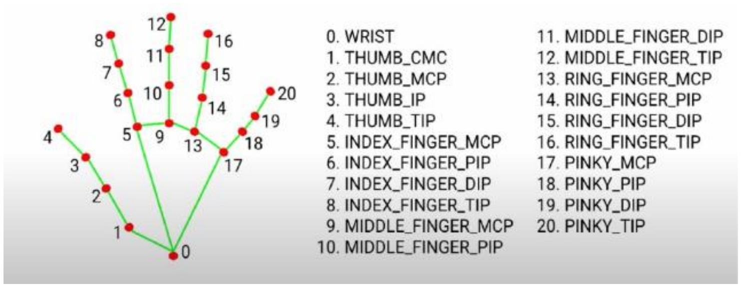
8.2 Detailed Design

Hand tracking is a process that uses computer vision to detect and focus on a hand's movement and orientation in an input image. This capability facilitates the development of numerous programs utilizing hand motion as input. The implementation requires writing code for hand tracking, which can later be modularized for reuse in various projects.

MediaPipe is utilized to simplify the hand-tracking process, breaking it into two stages:

1. **Palm Detection**: MediaPipe processes the input image, focusing on the hand, and generates a cropped image.
2. **Hand Landmark Identification**: MediaPipe identifies 21 key landmarks on the hand, enabling hand tracking and movement recognition.

These stages ensure accurate hand-tracking functionality while enabling modular integration into other projects.



Chapter 9
SECURITY FEATURES

__ ***Login Verification** (verify_login):*

- The verify_login function checks if the entered username and password exist in the user_db dictionary. If valid, it returns True; otherwise, False.

__ ***Login Action** (on_login):*

- Retrieves input from the login form and calls verify_login.
- On success, shows a "Login Successful" message and closes the application window.
- On failure, displays an error message indicating an invalid login.

__ ***User Registration** (register_user):*

- Collects data such as username, password, confirmation password, phone number, email, and CAPTCHA from the registration form.
- Performs validations:
    - Phone number must be 10 digits.
    - Email should follow a valid format.
    - CAPTCHA must match.
    - Password and confirm password must match.
    - Username should not already exist in user_db.
- If validation passes, adds the user to the user_db dictionary and switches back to the login form.

__ ***CAPTCHA Generation** (generate_captcha):*

- Generates a 5-digit random CAPTCHA using numbers (random.choices).
- Updates the CAPTCHA label with the generated code for the user to input.

__ ***Navigating Between Forms** (go_to_login):*

- The function hides the registration form and displays the login form.

__ ***Graphical User Interface (GUI)**:*

- Built using **Tkinter**, the interface consists of two frames: **Login Frame** and **Registration Frame**.
- Both frames have labels, text entry fields, and buttons for user interaction.

__ ***Login Frame**:*

- Contains fields for username and password.
- Buttons:
    - "Login" triggers the on_login function.

  - "Register" switches to the registration form.

__ ***Registration Frame***:

- Collects user details: Username, Password, Confirm Password, Phone Number, Email, and CAPTCHA.
- Validates input before registering the user.
- Buttons:
  - "Generate CAPTCHA" to generate and display a new CAPTCHA.
  - "Register" triggers the register_user function.
  - "Back to Login" returns to the login frame.

__ ***CAPTCHA Display***:

- CAPTCHA is displayed using a white-background label in the registration form.
- Users must input the correct CAPTCHA to complete registration.

__ ***Dynamic Form Switching***:

- Buttons and pack_forget()/pack() methods are used to dynamically switch between the login and registration forms.

__ ***Error Handling***:

- Appropriate error messages (using messagebox.showerror) are shown for invalid inputs or mismatched data, ensuring a user-friendly experience.

__ ***Success Messages***:

- Success messages (using messagebox.showinfo) are displayed for successful login and registration.

__ ***Data Storage***:

- User data (username and password) is stored in the user_db dictionary for verification purposes.

__ ***Secure Password Handling***:

- Password fields in both login and registration forms are obscured with characters (show="*" or show="&") for security.

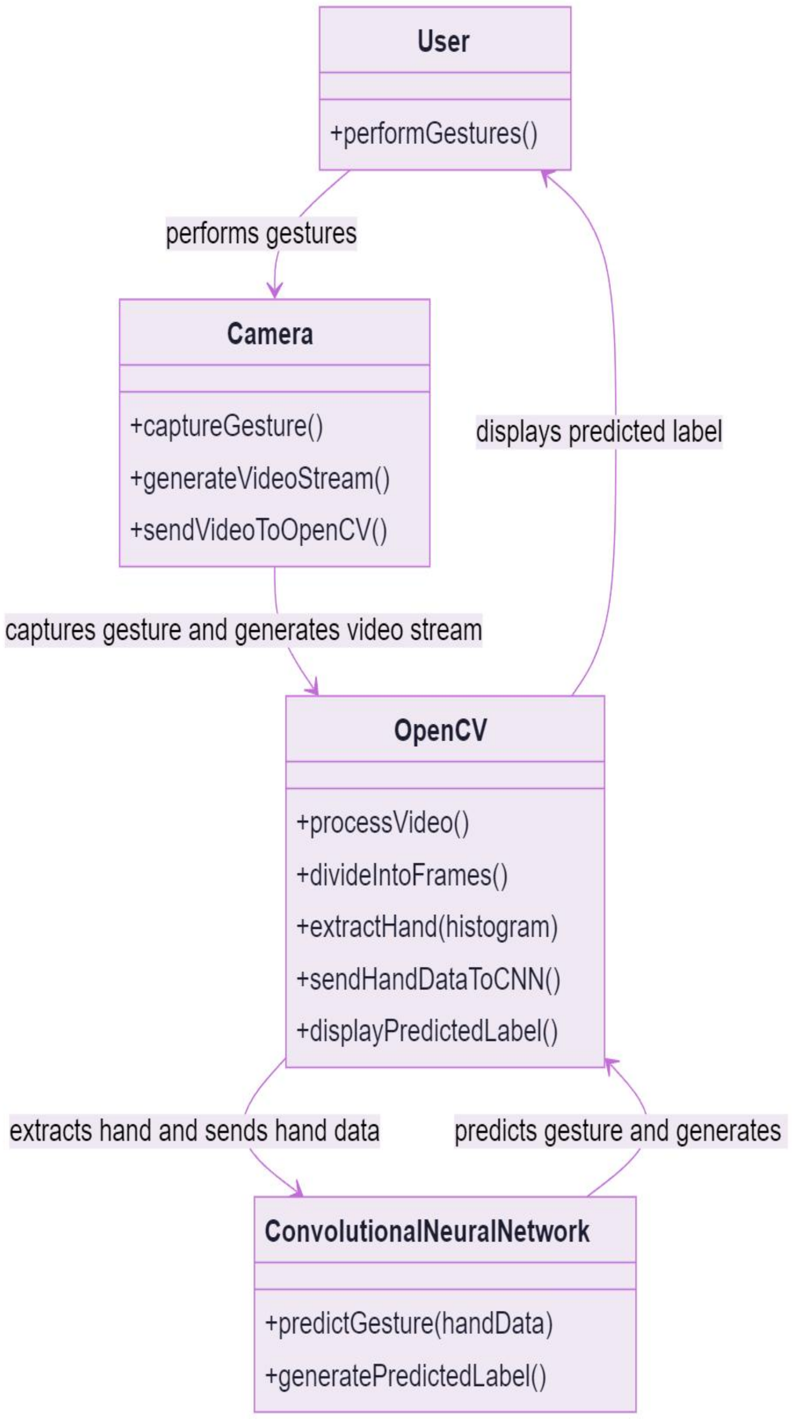
__ ***Responsive Design***:

- GUI elements are aligned using pack() with spacing (pady) for a neat layout.

__ ***Application Flow:***

- The app initializes with the **Login Frame**. Users can navigate to the **Registration Frame**, register successfully, and return to the login form for authentication.

Chapter 10
WORKFLOW DIAGRAM



GANTT CHART

Weeks Tasks	1-2	2-3	3-4	4-5	5-6	6-7	7-8	8-9	9-10
Feasible study	🟥	🟥							
Requirement analysis			🟨	🟨					
System design				🟩	🟩	🟩			
Coding						🟩	🟩		
Training and Testing								🟦	
Acceptance Testing								🟦	🟦
Maintenance	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦

Chapter 11
TESTING

11.1 Unit Testing

Test Case No.	**Feature**	**Input/Action**	**Expected Behavior**	**PASS/FAIL**
1	**Finger Detection**	Move finger close to the screen without touching	System should detect finger movement and allow drawing	PASS
2	**Brush Size Adjustment**	Change brush size to "Large" and draw	The system should draw with the selected larger brush size	PASS
3	**Color Palette Selection**	Choose a custom color from the palette	The selected color should be applied to the drawing	PASS
4	**Infinite Board**	Scroll to an empty area of the board	The canvas should extend seamlessly to accommodate more drawings	PASS
5	**Object Recognition**	Draw a rough circle	The system should recognize and correct the rough circle to a smooth one	PASS
6	**Image Recognition**	Insert an image of a car	The system should recognize the car and provide a suggestion or tagging	PASS
7	**Speech-to-Text Conversion**	Speak "Hello, FloatArt!"	The spoken text should appear as typed text on the board	PASS
8	**OpenAI Integration**	Perform a search query "What is AI?"	The system should display the top results for the query	PASS
9	**Undo Functionality**	Draw a shape and then press the Undo button	The last action (shape drawn) should be undone	PASS
10	**Redo Functionality**	Undo a shape and then press the Redo button	The undone shape should reappear	PASS
11	**Save Drawing**	Save the current drawing to the device	The file should be saved successfully in the specified format (e.g., PNG or JPEG)	PASS
12	**Load Drawing**	Load a previously saved drawing	The system should open the selected drawing	PASS
13	**Sign-In**	Enter valid credentials to sign in	User should successfully sign in and access their profile	PASS
14	**Register**	Enter user details (name, email, password) and complete the registration process	User account should be created successfully	PASS
15	**Captcha Validation**	Attempt login with a valid captcha	The system should validate the captcha and proceed with the login	PASS
16	**OTP Verification**	Enter an OTP during registration	The system should validate	PASS

		or login	the OTP and proceed	
17	**Logout**	Click on the logout button	User should be logged out and redirected to the login page	PASS
18	**Password Recovery**	Attempt to recover a password by entering a registered email	The system should send a password recovery email	PASS
19	**Eraser Tool**	Use the eraser tool to remove part of a drawing	The specified part of the drawing should be erased	PASS
20	**Help/Instructions**	Click on the "Help" button	The system should display a help menu or instructions for using the platform	PASS

11.2 Integration Testing

Case No.	**Modules Tested**	**Input/Action**	**Expected Behavior**	**Pass/Fail**
1	**Login, Captcha Validation**	Enter valid credentials with captcha	User should be logged in successfully after captcha validation	PASS
2	**Register, OTP Verification**	Register with valid details and enter the OTP	The system should create the account after OTP verification	PASS
3	**Speech-to-Text, Object Recognition**	Speak "Draw a star" and draw a rough star shape	The system should transcribe the speech, recognize the object, and auto-correct the rough star	PASS
4	**Save Drawing, Load Drawing**	Save a drawing and then reload it	The drawing should be saved and reloaded without any data loss	PASS
5	**Finger Detection, Brush Size**	Use finger detection to draw and change the brush size to "Small"	The system should allow drawing with the finger and apply the selected brush size	PASS
6	**Image Recognition, Background Tool**	Insert an image of a tree and select "Sky" as the background	The system should recognize the tree and successfully apply the sky background	PASS
7	**Undo, Redo**	Draw a circle, undo it, and redo the action	The circle should disappear on undo and reappear on redo	PASS
8	**OpenAI Integration, Speech-to-Text**	Speak "Search for latest AI trends"	The system should transcribe the speech and perform the search, displaying top results	PASS
9	**Multi-touch, Infinite Board**	Use two fingers to draw on an extended part of the board	The system should detect both touches and allow drawing on the extended canvas	PASS
10	**Logout, Login**	Log out and log back in with valid credentials	The system should log out successfully and allow the user to log back in without errors	PASS

Chapter 12
IMPLEMENTATION

Module/Library	Purpose
random	Generates random numbers or values
cv2 (OpenCV)	Image and video processing
numpy	Supports multi-dimensional arrays and numerical operations
speech_recognition	Converts speech to text
handTracker (Custom Module)	Detects and tracks hand gestures
requests	Makes HTTP requests
bs4 (BeautifulSoup)	Parses HTML/XML documents for web scraping
tkinter (as tk)	Creates the graphical user interface (GUI)
messagebox	Displays popup messages
smtplib	Sends emails
email.mime.text	Constructs plain-text email messages
email.mime.multipart	Builds emails with multiple sections or attachments
re	Provides regular expression support
mediapipe	Provides ML solutions for tasks like pose estimation, face mesh, and hand tracking
pandas	Data analysis and manipulation tool
TensorFlow	Open-source machine learning framework for building and training models

Chapter 13
CODING STANDARDS

Adopting consistent coding standards ensures maintainability, readability, and efficiency throughout the project. Below are the coding standards tailored for the FloatArt project:

1. General Guidelines

- **Consistent Formatting**: Use a consistent indentation style (e.g., 4 spaces for Python).
- **Meaningful Names**: Use descriptive variable, function, and class names (draw_circle instead of dc).
- **Comments and Documentation**:
  - Include docstrings for functions and classes using the Google or NumPy style.
  - Add inline comments for complex logic but avoid over-commenting trivial code.
  - Include a header comment in each file describing its purpose.

2. Code Structure

- **Modularity**: Break down the project into smaller, reusable functions and modules.
- **Separation of Concerns**: Separate UI, business logic, and backend services into different files or layers.
- **Error Handling**: Include proper exception handling (try-except blocks) to ensure graceful error recovery.
- **Configuration**: Store constants, configurations, and API keys in separate configuration files (e.g., config.py).

3. Naming Conventions

- **Variables**: Use snake_case for variable and function names (color_palette, capture_speech).
- **Classes**: Use PascalCase for class names (DrawingTool, ImageProcessor).
- **Constants**: Use UPPER_CASE for constants (MAX_BRUSH_SIZE, DEFAULT_COLOR).

4. Coding Practices

- **Avoid Hardcoding**: Use configuration files or environment variables for constants like URLs, API keys, or colors.
- **Limit Line Length**: Keep lines under 80-100 characters to improve readability.
- **Optimize Loops**: Avoid nested loops where possible; use list comprehensions for brevity and performance.
- **Avoid Global Variables**: Pass variables as parameters or use encapsulation in classes.

5. Security

- **Input Validation**: Validate all user inputs, especially for critical features like login, registration, and OTP.
- **Secure API Calls**: Use HTTPS and store API keys securely (e.g., environment variables, .env files).
- **Captcha & OTP**: Implement robust mechanisms for CAPTCHA validation and OTP expiry.
- **Authentication**: Use encrypted storage for passwords and tokens.

6. Version Control

- **Commit Messages**: Use descriptive commit messages (e.g., Added infinite board scrolling feature).
- **Branching**: Use feature-specific branches (feature/integration) and merge into the main branch after review.
- **Pull Requests**: Code changes must undergo peer review before merging.

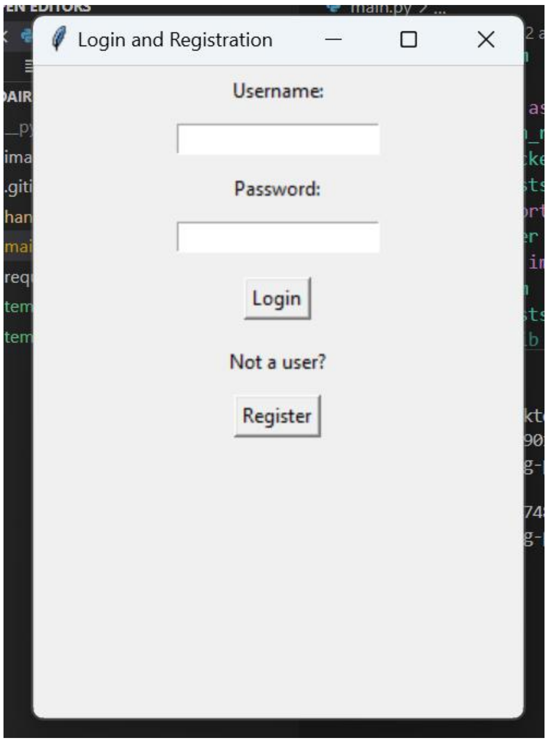
7. Tools and Frameworks

- **Linting**: Use flake8 or pylint for Python to enforce style rules.
- **Versioning**: Follow semantic versioning (MAJOR.MINOR.PATCH) for project releases.
- **Documentation**: Use tools like Sphinx or MkDocs for generating detailed project documentation.

8. Performance Optimization

- **Asynchronous Operations**: Use asynchronous functions for network requests (e.g., asyncio for Python).
- **Caching**: Cache frequent operations like API responses to minimize latency.
- **Profiling**: Use performance profiling tools like cProfile to identify bottlenecks.

Chapter 14
RESULTS & SNAPSHOTS



Login and Registration

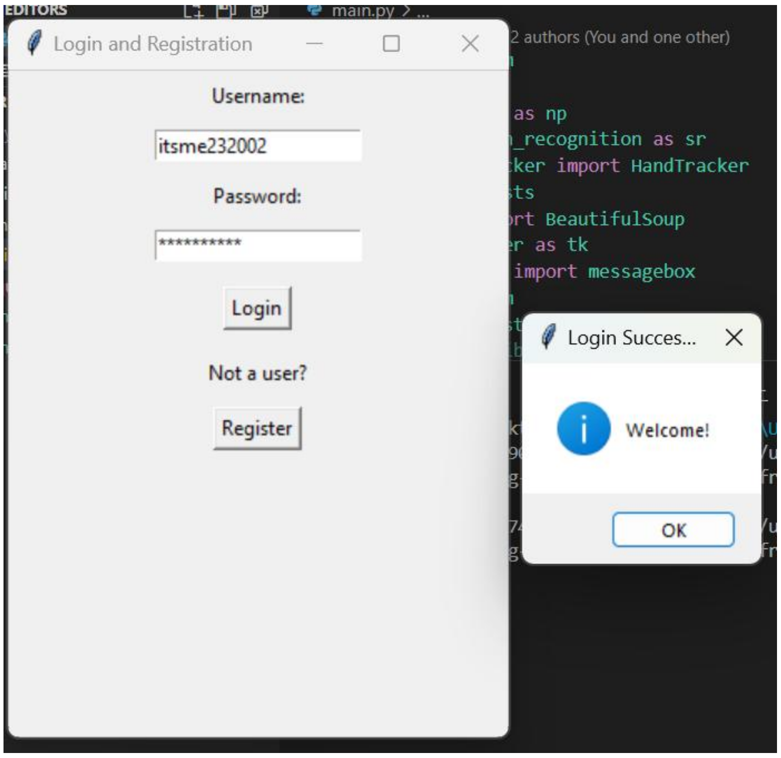
Username:

Password:

Login

Not a user?

Register



Login and Registration

Username: itsme232002

Password: **********

Login

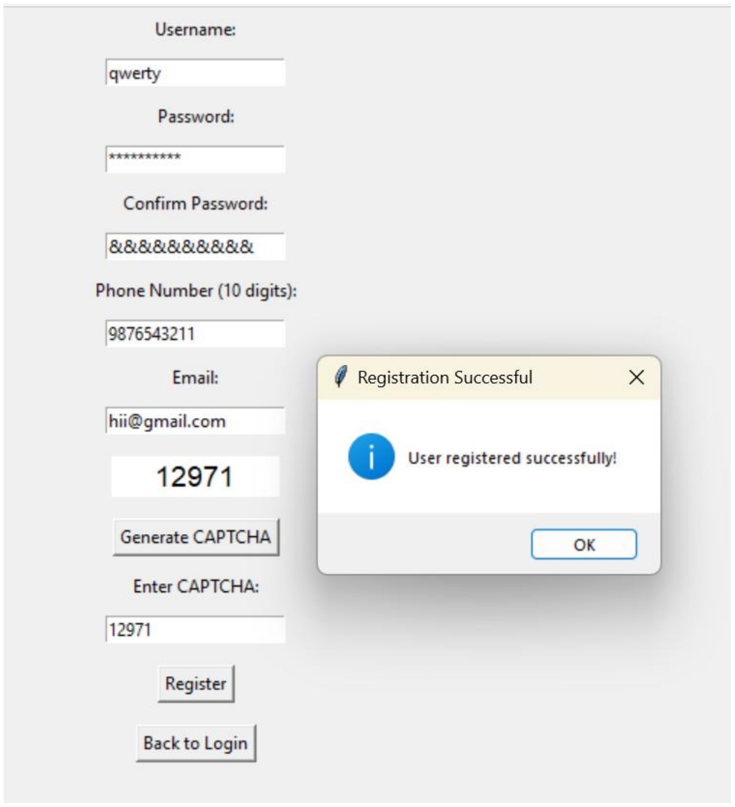
Not a user?

Register

Login Succes...

Welcome!

OK



Username: qwerty

Password: **********

Confirm Password: &&&&&&&&&&

Phone Number (10 digits): 9876543211

Email: hii@gmail.com

12971

Generate CAPTCHA

Enter CAPTCHA: 12971

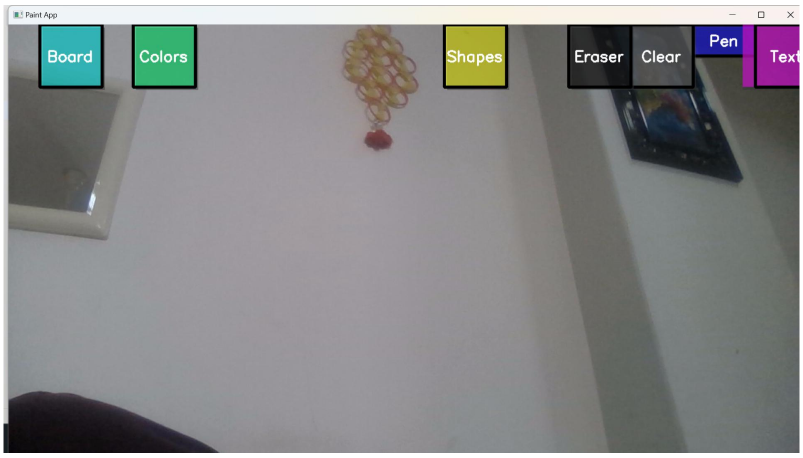
Register

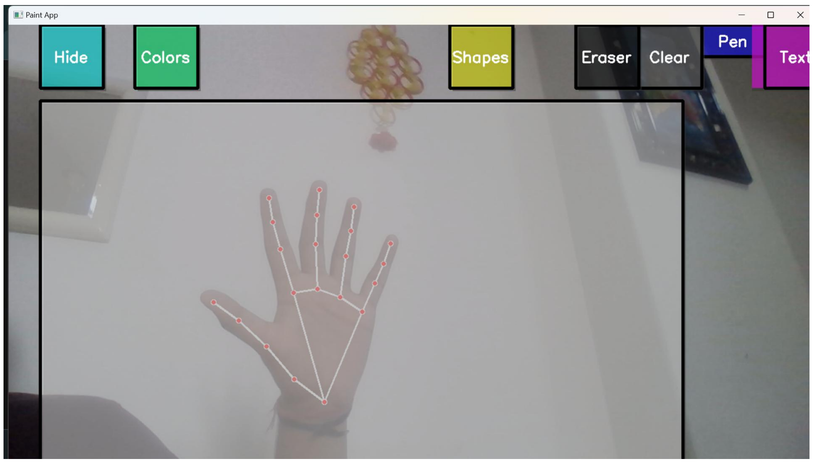
Back to Login

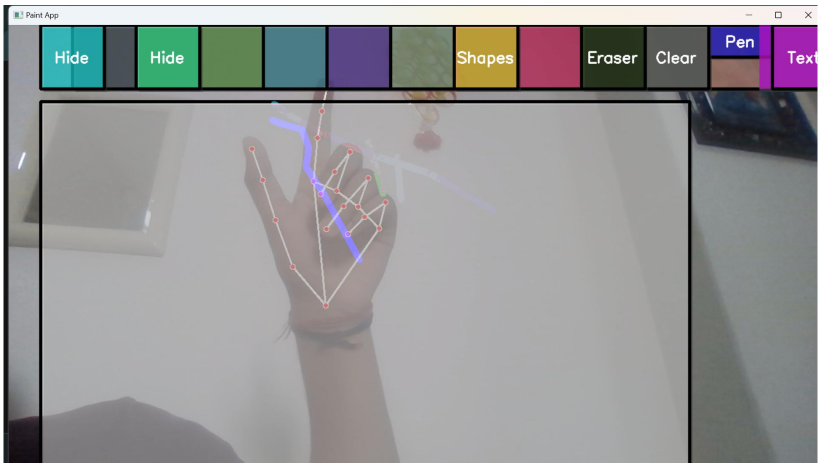
Registration Successful

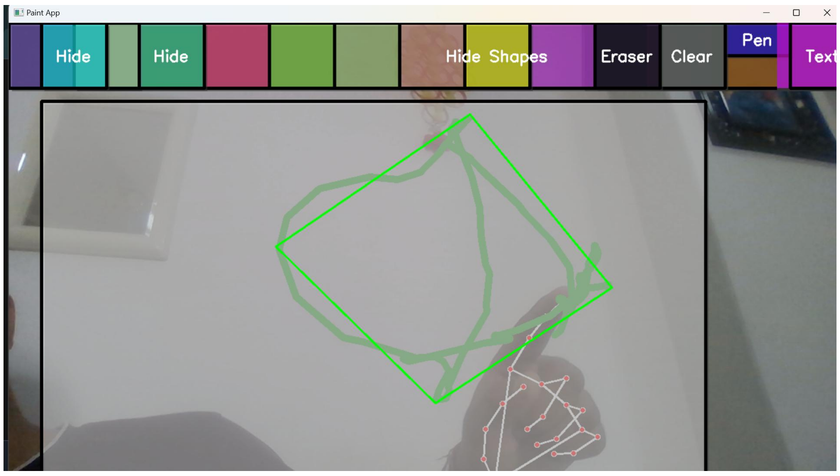
User registered successfully!

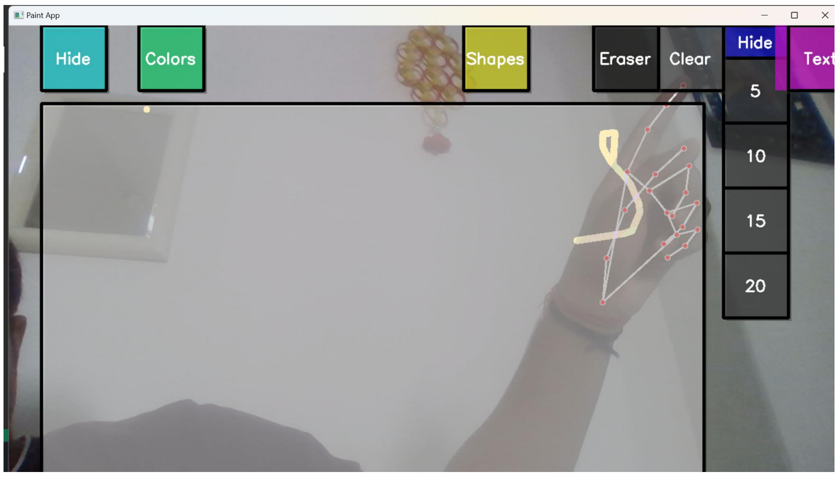
OK

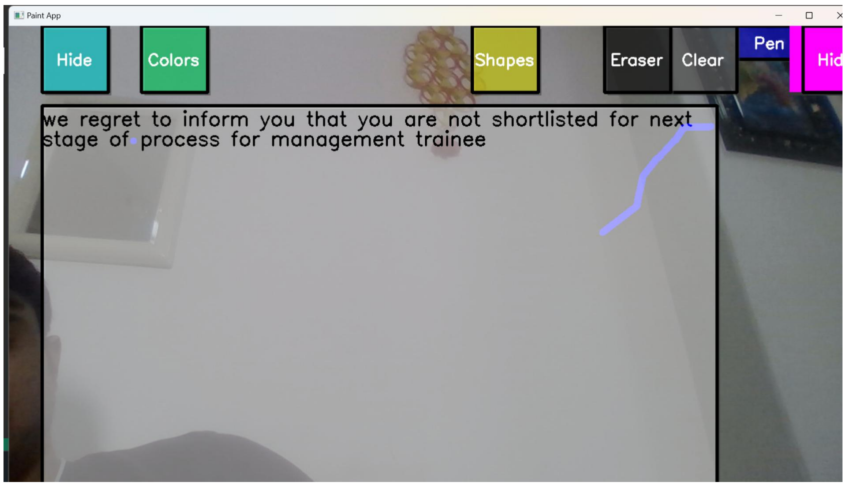

Paint App
Board
Colors
Shapes
Eraser
Clear
Pen
Text


Paint App
Hide
Colors
Shapes
Eraser
Clear
Pen
Text

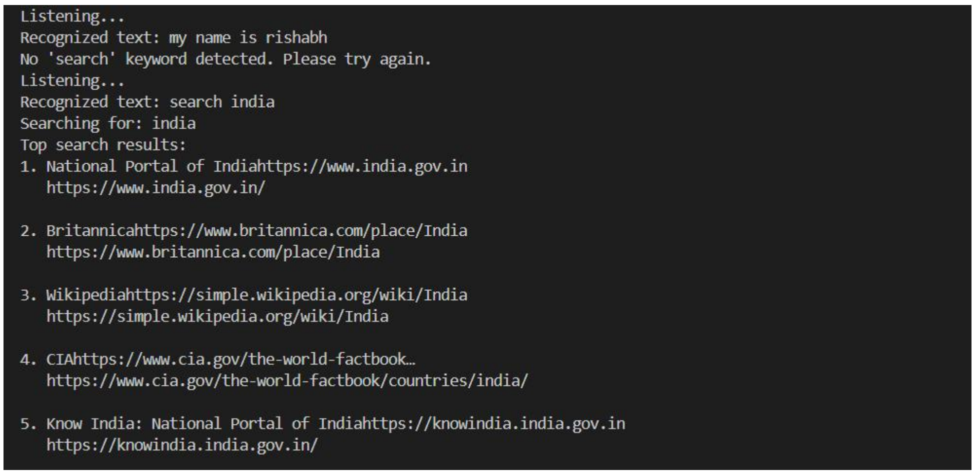

Paint App
Hide
Hide
Shapes
Eraser
Clear
Pen
Text







we regret to inform you that you are not shortlisted for next stage of process for management trainee



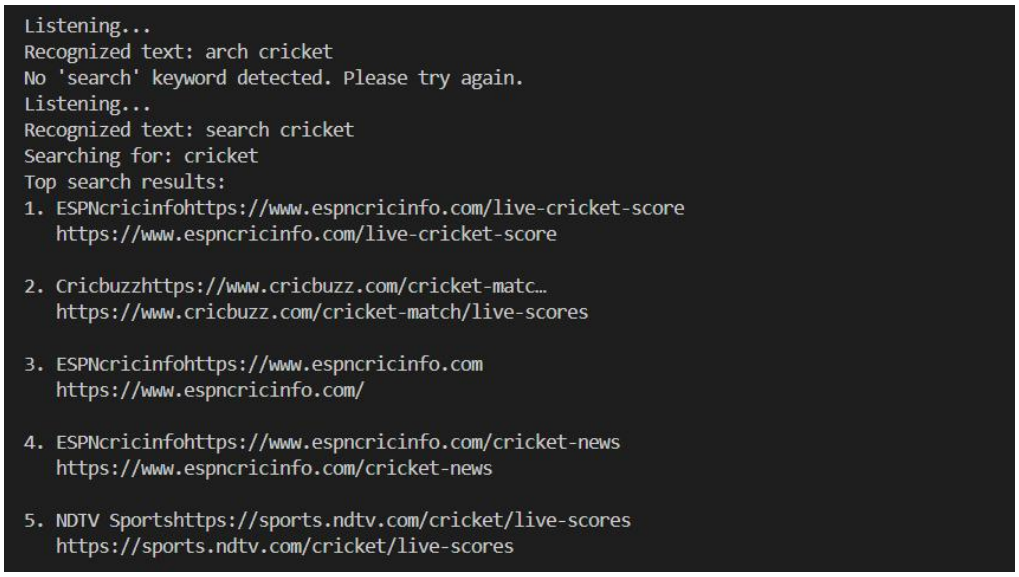
```
Listening...
Recognized text: my name is rishabh
No 'search' keyword detected. Please try again.
Listening...
Recognized text: search india
Searching for: india
Top search results:
1. National Portal of Indiahttps://www.india.gov.in
   https://www.india.gov.in/

2. Britannicahttps://www.britannica.com/place/India
   https://www.britannica.com/place/India

3. Wikipediahttps://simple.wikipedia.org/wiki/India
   https://simple.wikipedia.org/wiki/India

4. CIAhttps://www.cia.gov/the-world-factbook…
   https://www.cia.gov/the-world-factbook/countries/india/

5. Know India: National Portal of Indiahttps://knowindia.india.gov.in
   https://knowindia.india.gov.in/
```



```
Listening...
Recognized text: arch cricket
No 'search' keyword detected. Please try again.
Listening...
Recognized text: search cricket
Searching for: cricket
Top search results:
1. ESPNcricinfohttps://www.espncricinfo.com/live-cricket-score
   https://www.espncricinfo.com/live-cricket-score

2. Cricbuzzhttps://www.cricbuzz.com/cricket-matc…
   https://www.cricbuzz.com/cricket-match/live-scores

3. ESPNcricinfohttps://www.espncricinfo.com
   https://www.espncricinfo.com/

4. ESPNcricinfohttps://www.espncricinfo.com/cricket-news
   https://www.espncricinfo.com/cricket-news

5. NDTV Sportshttps://sports.ndtv.com/cricket/live-scores
   https://sports.ndtv.com/cricket/live-scores
```

Chapter 15
FUTURE SCOPE OF STUDY

✓ · ***Integration of Advanced AI Features***

- **Real-time Hand Gesture Recognition**: Enhance the system to recognize complex gestures like zoom, rotate, or multi-touch inputs for precise control.
- **AI-powered Art Suggestions**: Implement machine learning models to suggest improvements or additional elements based on the user's drawing style.
- **Generative AI**: Incorporate AI models like DALL-E to auto-generate images or enhance user-created drawings.

✓ · ***Enhanced Collaboration and Accessibility***

- **Real-time Multi-user Collaboration**: Enable multiple users to work on the same virtual board simultaneously, with features like role-based access (viewer, editor, etc.).
- **Cross-platform Integration**: Expand accessibility to mobile devices, tablets, and web browsers for a seamless experience across platforms.
- **Language Support**: Introduce multilingual speech-to-text and text-to-speech capabilities to cater to a global audience.

✓ · ***Improved Security and Data Management***

- **Cloud Storage Integration**: Provide secure cloud storage options for saving and retrieving projects, ensuring data persistence and accessibility.
- **Enhanced User Authentication**: Implement biometric authentication (e.g., fingerprint or face recognition) for secure login and access.
- **Data Analytics Dashboard**: Offer analytics for users to track project progress, time spent, and collaboration insights.

REFERENCES

[1] Saira Beg, M. Fahad Khan and Faisal Baig, "Text Writing in Air," Journal of Information Display Volume 14, Issue 4, 2013

[2] Alper Yilmaz, Omar Javed, Mubarak Shah, "Object Tracking: A Survey", ACM Computer Survey. Vol. 38, Issue. 4, Article 13, Pp. 1-45, 2006

[3] Yuan-Hsiang Chang, Chen-Ming Chang, "Automatic Hand-Pose Trajectory Tracking System Using Video Sequences", INTECH, pp. 132- 152, Croatia, 2010

[4] Erik B. Sudderth, Michael I. Mandel, William T. Freeman, Alan S. Willsky, "Visual Hand Tracking Using Nonparametric Belief Propagation", MIT Laboratory For Information & Decision Systems Technical Report P-2603, Presented at IEEE CVPR Workshop On Generative Model-Based Vision, Pp. 1-9, 2004

[5] T. Grossman, R. Balakrishnan, G. Kurtenbach, G. Fitzmaurice, A. Khan, and B. Buxton, "Creating Principal 3D Curves with Digital Tape Drawing," Proc. Conf. Human Factors Computing Systems (CHI' 02), pp. 121- 128, 2002.

[6] T. A. C. Bragatto, G. I. S. Ruas, M. V. Lamar, "Real-time Video-Based Finger Spelling Recognition System Using Low Computational Complexity Artificial Neural Networks", IEEE ITS, pp. 393-397, 2006

INDIVIDUAL CONTRIBUTION REPORT:

“ FloatArt: LET YOUR IMAGINATION TAKE A VIRTUAL FLIGHT! “

AKASH CHANDRAKAR

Abstract: FloatArt is a trailblazing virtual jamboard designed to revolutionize digital creative environments by leveraging the power of machine learning (ML), deep learning (DL), and artificial intelligence (AI). This innovative platform offers a touchless interface that enables users to engage in intuitive drawing, writing, and collaborative tasks through advanced hand gesture recognition technology.

The system boasts a wide range of features, including infinite color palettes, precise shape correction, and customizable brush sizes, catering to diverse creative needs. It integrates speech-to-text recognition, enabling seamless transcription of spoken words into written text, and incorporates OpenAI-powered internet search to provide real-time knowledge and insights. By analyzing user inputs in real time, FloatArt refines shapes and gestures, ensuring unparalleled precision and usability in digital workspaces. This platform is designed to enhance creative and collaborative efforts, making it particularly valuable for artists, designers, and educators. FloatArt redefines virtual collaboration by providing a versatile, high-tech solution that empowers users to explore their creativity while boosting productivity and efficiency in digital environments.

Individual contribution and findings: As a key contributor to the FloatArt project, I was responsible for designing and developing the UI/UX, the color palette system, and managing the project’s state. My work on the UI/UX focused on creating an intuitive and user-friendly interface tailored to the needs of a virtual jamboard application. I ensured that the interface was clean and minimalistic, providing users with easily navigable toolbars, draggable panels, and resizing options. Special attention was given to responsiveness, ensuring seamless usability across devices with varying screen sizes. While designing the color palette system, I implemented an infinite selection mechanism, allowing users to customize and save preferred colors effortlessly. Additionally, I managed the project’s state using effective state management techniques, ensuring smooth transitions, real-time updates, and synchronization of changes across various components.

Individual contribution to project report preparation: I contributed to the project report by conducting literature reviews and preparing base abstracts, which helped identify relevant technologies and industry standards. This research informed the development of FloatArt, highlighting key insights and aligning with project goals. Additionally, I was responsible for testing the system, developing test cases for functionality and integration, and identifying bugs and performance issues. My testing efforts were crucial for refining the project and ensuring a high-quality final product.

Individual contribution for project presentation and demonstration: I have taken the lead in crafting the data flows for the app, producing both detailed PowerPoint (PPT) presentations and illustrative diagrams. In this capacity, my role involves visually mapping out the intricate data pathways within the application, ensuring a clear representation of information transfer and processes. The PPT presentations are designed to effectively communicate these data flows, providing a comprehensive overview for both technical and non-technical stakeholders. Through the creation of these diagrams, my objective is to enhance understanding and facilitate seamless communication regarding the intricate data architecture of the app, contributing to the overall clarity and effectiveness of the project.

Full Signature of Supervisor: Full signature of the student:

INDIVIDUAL CONTRIBUTION REPORT:

“ FloatArt: LET YOUR IMAGINATION TAKE A VIRTUAL FLIGHT! “

RISHABH MOHATA

Abstract: FloatArt is a trailblazing virtual jamboard designed to revolutionize digital creative environments by leveraging the power of machine learning (ML), deep learning (DL), and artificial intelligence (AI). This innovative platform offers a touchless interface that enables users to engage in intuitive drawing, writing, and collaborative tasks through advanced hand gesture recognition technology.

The system boasts a wide range of features, including infinite color palettes, precise shape correction, and customizable brush sizes, catering to diverse creative needs. It integrates speech-to-text recognition, enabling seamless transcription of spoken words into written text, and incorporates OpenAI-powered internet search to provide real-time knowledge and insights. By analyzing user inputs in real time, FloatArt refines shapes and gestures, ensuring unparalleled precision and usability in digital workspaces. This platform is designed to enhance creative and collaborative efforts, making it particularly valuable for artists, designers, and educators. FloatArt redefines virtual collaboration by providing a versatile, high-tech solution that empowers users to explore their creativity while boosting productivity and efficiency in digital environments.

Individual contribution and findings: In the **FloatArt** project, I was responsible for developing the core logic, key functionalities, and integrating OpenAI’s system. I designed the structure of the project, ensuring modularity and scalability, and implemented features like speech-to-text, color selection, brush size adjustments, background changes, and auto-correction of drawn objects. I also integrated multi-touch functionality for intuitive interaction. I imported necessary dependencies such as opencv, Pillow, speech_recognition, and openai to handle tasks like image manipulation, object recognition, and speech-to-text conversion. Additionally, I incorporated OpenAI’s GPT models to enable voice-activated web searches, enhancing user interaction. Through this, I learned the importance of real-time feedback in interactive systems and how to optimize performance for smooth, concurrent user interactions.

Individual contribution to project report preparation: I contributed to the **FloatArt** project report by outlining the proposed model, which included features like speech-to-text, object recognition, and OpenAI integration for voice-activated web searches. I documented the methodologies used, such as Agile development, machine learning, and speech recognition. I also covered integration testing, ensuring smooth interaction between the software and hardware, and provided hardware and software specifications, including recommendations for equipment like a touchscreen and microphone, as well as libraries. These contributions ensured a comprehensive and clear project report.

Individual contribution for project presentation and demonstration: I have taken on the comprehensive responsibility of preparing the entire PowerPoint (PPT) presentation. This involves the end-to-end process of conceptualization, content creation, and design implementation. I have meticulously structured the presentation to convey key messages effectively, ensuring a logical flow and visual appeal. From outlining the main points to selecting appropriate visuals, each aspect of the PPT has been crafted with precision to engage and inform the audience. By overseeing the entirety of the PPT creation process, I aim to deliver a cohesive and impactful presentation that aligns seamlessly with the objectives and expectations of the project or assignment.

Full Signature of Supervisor: Full signature of the student:

INDIVIDUAL CONTRIBUTION REPORT:

" FloatArt: LET YOUR IMAGINATION TAKE A VIRTUAL FLIGHT! "

TUSHAR BHATT

Abstract: FloatArt is a trailblazing virtual jamboard designed to revolutionize digital creative environments by leveraging the power of machine learning (ML), deep learning (DL), and artificial intelligence (AI). This innovative platform offers a touchless interface that enables users to engage in intuitive drawing, writing, and collaborative tasks through advanced hand gesture recognition technology.

The system boasts a wide range of features, including infinite color palettes, precise shape correction, and customizable brush sizes, catering to diverse creative needs. It integrates speech-to-text recognition, enabling seamless transcription of spoken words into written text, and incorporates OpenAI-powered internet search to provide real-time knowledge and insights. By analyzing user inputs in real time, FloatArt refines shapes and gestures, ensuring unparalleled precision and usability in digital workspaces. This platform is designed to enhance creative and collaborative efforts, making it particularly valuable for artists, designers, and educators. FloatArt redefines virtual collaboration by providing a versatile, high-tech solution that empowers users to explore their creativity while boosting productivity and efficiency in digital environments.

Individual contribution and findings: In the **FloatArt** project, I developed the CAPTCHA code base to enhance security by generating visual challenges for users. I maintained coding standards throughout the project, ensuring clean, readable, and maintainable code. I also focused on creating a well-organized code structure, enabling modular development and scalability. Additionally, I defined intuitive and visually appealing color schemes for the virtual jamboard, enhancing user experience. These contributions helped ensure a secure, efficient, and user-friendly system, while maintaining a focus on clean code and effective design.

Individual contribution to project report preparation: In the preparation of the FloatArt project report, I contributed to creating flow diagrams, detailing the system's architecture and user interactions. I documented the security features, highlighting the CAPTCHA integration to prevent unauthorized access and ensure user data protection. Additionally, I explained the methodologies used in the project, including Agile development, speech recognition, and machine learning integration for object recognition and auto-correction. These contributions helped in clearly presenting the project's design, security measures, and technical approach in the report.

Individual contribution for project presentation and demonstration: I have taken on the comprehensive responsibility of preparing the entire PowerPoint (PPT) presentation. This involves the end-to-end process of conceptualization, content creation, and design implementation. I have meticulously structured the presentation to convey key messages effectively, ensuring a logical flow and visual appeal. From outlining the main points to selecting appropriate visuals, each aspect of the PPT has been crafted with precision to engage and inform the audience. By overseeing the entirety of the PPT creation process, I aim to deliver a cohesive and impactful presentation that aligns seamlessly with the objectives and expectations of the project or assignment.

Full Signature of Supervisor: Full signature of the student: