PROFORMA FOR THE APPROVAL PROJECT PROPOSAL

PNR **No.:2021016400920152** Roll no**: 753**_________

1. Name of the Student

   Faiyaz Zuber Maknojia____

2. Title of the Project

   AI VIRTUAL PAINTER____

3. Name of the Guide

   Ms Sonal Chavan____

4. Teaching experience of the Guide____

5. Is this your first submission? Yes □ No □

Signature of the Student Signature of the Guide

Date: ………………… Date: …………………….

Signature of the

Coordinator

Date:

………………

AI VIRTUAL PAINTER

A Project Report
Submitted in partial fulfillment of the
Requirements for the award of the Degree
of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

By

Maknojia Faiyaz Zuber Hafsa

A753

Under the esteemed guidance of

Ms Sonal Chavan

DEPARTMENT OF INFORMATION TECHNOLOGY

CHIKITSAK SAMUHA'S
S.S & L.S PATKAR COLLEGE OF ARTS & SCIENCE & V. P. VARDE COLLEGE OF COMMERCE & ECONOMICS.
An Autonomous College
Affiliated To University Of Mumbai
Goregaon (W), Mumbai – 400 062

CHIKITSAK SAMUHA’S

S.S & L.S PATKAR COLLEGE OF ARTS & SCIENCE & V. P. VARDE COLLEGE OF COMMERCE & ECONOMICS.

An Autonomous College

DEPARTMENT OF INFORMATION TECHNOLOGY

<u>CERTIFICATE</u>

This is to certify that the project entitled, **"AI VIRTUAL PAINTER"**, is bonafide work of **Mr. Faiyaz Zuber Maknojia** bearing Seat.No:**A753** submitted in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY fromUniversity of Mumbai.

Internal Guide **Coordinator**

External Examiner

Date: **CollegeSeal**

ABSTRACT

The **AI Virtual Painter** is a system that allows users to paint virtually using hand gestures. The system uses computer vision and machine learning to track the user's hand movements and translate them into brush strokes. This allows users to draw freely and expressively, without the need for a physical paintbrush.The AI Virtual Painter has a number of potential benefits. It can be used to improve creativity and artistic expression. It can also be used to provide a more engaging and interactive learning experience for students of art. Additionally, the AI Virtual Painter can be used to create virtual paintings that can be shared with others online.The objective of the AI Virtual Painter is to develop a system that is easy to use, accurate, and versatile. The system should be able to track a wide range of hand gestures and translate them into smooth and natural brush strokes. The system should also be able to support a variety of painting styles, from realistic to abstract.The AI Virtual Painter is still under development, but it has the potential to revolutionize the way people paint. It is a powerful tool that can be used to create beautiful and expressive art.The AI Virtual Painter project is a collaborative effort between researchers from computer science, artificial intelligence, and art. We believe that the AI Virtual Painter has the potential to make a significant impact on the way people create and experience art.

ACKNOWLEDGEMENT

I would like to extend my sincere appreciation to the Information Technology department of Patkar-Varde College for their invaluable support in guiding me through the completion of my project. Finally, I am delighted to present my project after months of dedicated effort. The journey of creating this project was filled with novel experiences, extensive learning, and various challenges. Despite its difficulty, the prompt assistance I received made it manageable and helped turn my ideas into reality.I am truly grateful to everyone who played a role in assisting me with the successful completion of my project. My heartfelt thanks go to my family and friends, who have consistently been supportive and encouraging throughout all my endeavors. Special thanks also go to our CEO**, Dr. (Mrs.) Mala Kharkar**, the Principal, **Dr. Pratibha Gaikwad**, the B.Sc.I.T. coordinator, **Namrata Kawle,** and the co-coordinator**, Mr. Chayan Bhattacharjee**, for providing me with the necessary facilities.I cannot overlook the invaluable guidance of **Ms. Sonal Chavan,** my project advisor, whose encouragement and support were instrumental in the successful development of my research. Without her guidance, completing this project would not have been possible. She played a crucial role in helping me stay focused and committed to the execution of the project. I am also indebted to my friends for their unwavering support during the course of my project

DECLARATION

I hereby declare that the project entitled, “**AI Virtual Painter**” done at **Patkar Varde College**, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfillment of the requirements for the award of degree of **BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)** to be submitted as a final semester project as part of our curriculum.

Name and Signature of the Student

Date:

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 : INTRODUCTION

1. Theoretical Background

AI virtual painters are computer programs that use artificial intelligence (AI) to create paintings. They are based on a number of different areas of AI, including computer vision, machine learning, and natural language processing. AI virtual painters use these technologies to track the user's hand movements, identify objects and scenes, understand instructions, and generate text descriptions of paintings.AI virtual painter is constantly evolving as new advances are made in artificial intelligence. This is a rapidly growing field with a lot of potential, and it is exciting to see what the future holds for AI virtual painters.

2. Objectives of the Project

- Enable users to draw and paint on a digital canvas using their hands.
- Provide a variety of tools and features to create complex and expressive drawings.
- Make it easy for users to share their drawings with others.
- Be accessible to users of all ages and abilities.

3. Purpose, Scope & Applicability of the Project

a. Purpose

- To provide a new and innovative way for people to create art, regardless of their age, ability, or experience.
- The goal of an AI virtual painter is to make art more accessible and enjoyable for everyone. It is a powerful tool that can be used to express creativity and imagination, and to create something truly unique.

b. Scope

- Limitation
- Limited input methods: AI virtual painters typically require input from a mouse or keyboard. This can be limiting for artists who prefer to use other input methods, such as a touchscreen or a stylus pen.
- Lack of understanding of context: AI virtual painters do not have the same understanding of context as humans do. This can lead to problems when they are asked to paint something that is outside of their training data.
- Susceptibility to adversarial attacks: AI virtual painters can be fooled by adversarial attacks, which are carefully crafted inputs that can cause the AI to produce incorrect or unintended output.

- Accessibility

  Include voice control: This would allow users who are unable to use a mouse or keyboard to control the AI virtual painter with their voice.

  Support screen readers: This would allow users who are blind or visually impaired to use the AI virtual painter by reading aloud the text and images on the screen.

  Provide large print and high contrast options: This would make the text and images on the screen easier to see for users with visual impairments.

c. Applicability

- AI virtual painter project has the potential to be applicable to a wide range of industries and applications. It can be used to teach art, create digital art, collaborate with other people, entertain people, and even help people with disabilities. As AI virtual painters continue to develop, they are likely to become even more applicable to a wider range of industries and applications.

4. Expected Achievements

- Make art more accessible and expressive.
- Promote collaboration and creativity.
- Revolutionize the art industry.

5. Organization of Report

Chapter 1, Lays the foundation by discussing the theoretical background, project objectives, scope, applicability, and expected achievements, providing an overview of what the project entails.

Chapter 2, Briefly describes available technologies, compares them, and justifies the chosen technology for the project, providing insights into the decision-making process, resulting in the final selection and its supporting reasons.

Chapter 3, Addresses the project's problem definition, its requirements specifications, and comprehensive feasibility study. It further presents a detailed plan and schedule, utilizing Gantt and Pert charts. Additionally, various conceptual models, including Use-Case, Activity, Data-Flow, Class, E-R, and Sequence diagrams, are discussed to enhance project understanding and management.

Chapter 4, Explains our project with the help of a logic diagram, outlining its fundamental modules. The chapter also delves into the UI architecture. Additionally, it addresses potential security concerns and offers viable solutions for mitigation.

Chapter5, Implementation and Testing, Our implementation plan, standards, and protocols that were followed in implementing the AI VIRTUAL PAINTER are described in this chapter. Our most significant point in this chapter is about the testing approach we have taken after the completion of the project. Here we discussed functional testing, non-functional testing, scalability and etc. Once the testing was complete, we note everything down as test cases. Additionally, we found efficient coding and effective code.

Chapter6, Results and discussions, this chapter summarizes the test case we generated in the previous chapter as a test report. We also discuss the objectives behind the test cases. In addition, we drafted the user documentation for the user, as well as estimated the project's costs.

Chapter7, Conclusions, as the chapter name says in this chapter we talk about the conclusion of the AI VIRTUAL PAINTER. We also mention the limitations and future scope of AI VIRTUAL PAINTER.

CHAPTER 2 : SURVEY OF TECHNOLOGIES

1. Description of Available Technologies

- Computer Vision: Computer vision involves processing visual data from cameras to interpret and understand the surrounding environment. In the context of the game, computer vision algorithms analyze the video feed to detect and track the player's hand movements. This technology enables the system to recognize and translate gestures into in-game actions.
- Gesture Recognition: Gesture recognition involves identifying specific patterns in hand movements to determine the intended gesture. This technology employs machine learning models and algorithms to distinguish between different gestures, such as swipes, taps, and circles.
- VS Code: Visual Studio Code (VS Code) is a code editor developed by Microsoft for Windows, Linux, and macOS. It is a lightweight but powerful editor that can be used to code in a variety of programming languages, including JavaScript, Python, Java, C++, and C#.

2. Comparative Analysis of Technologies in Chosen Area

Features	Visual Studio Code	Sublime Text	Atom
Pricing	Free and open-source	Freemium (personal use is free, commercial use requires a license)	Free and open-source
Performance	Good performance, especially for large files	Very fast performance	Can be slow for large files
Features	Wide range of features, including code completion, linting, debugging, and integration with Git and other tools	Fewer features than VS Code, but still includes code completion, linting, and a built-in terminal	Similar to Sublime Text, but with a wider range of features and integration with Git and other tools
Customization	Highly customizable with extensions	Can be customized with plugins	Can be customized with plugins

Popularity	Very popular, especially among web developers	Less popular than VS Code, but still has a larger in use.	Less popular than VS Code and Sublime Text

3. Chosen Project Domain

1. AI

Artificial Intelligence (AI) is a branch of computer science focused on creating systems capable of performing tasks that typically require human intelligence. These tasks include learning, reasoning, problem-solving, perception, and natural language understanding. AI technologies range from machine learning to robotics, impacting various industries and daily life.

2. Images Processing

Image processing involves manipulating or analyzing images using algorithms and computational techniques. It aims to enhance, transform, or extract information from images. Applications include medical imaging, facial recognition, and improving image quality in photography. Techniques include filtering, edge detection, and image compression.

4. Technologies to be used

a. Backend: Python

b. Framework: Vs Code

5. Reason Supporting the use of above selected technologies

Visual Studio Code (VS Code) is a code editor developed by Microsoft for Windows, Linux, and macOS. It is a lightweight but powerful editor that can be used to code in a variety of programming languages, including JavaScript, Python, Java, C++, and C#.

VS Code is known for its flexibility and extensibility. It has a large number of extensions that can be added to add new features and functionality. This makes it a very versatile editor that can be used for a variety of tasks.

Here are some of the features of VS Code:

- Lightweight and fast: VS Code is a lightweight editor that starts up quickly and uses very little memory. This makes it a good choice for users who have older computers or who need to work on projects with large files.
- Flexible and extensible: VS Code is a very flexible editor that can be customized to meet the needs of individual users. It has a large number of extensions that can be added to add new

features and functionality.

- Multi-language support: VS Code supports a wide range of programming languages, including JavaScript, Python, Java, C++, and C#. This makes it a good choice for users who need to work on projects with multiple languages.
- Integrated debugger: VS Code has a built-in debugger that can be used to step through code and inspect variables. This makes it easy to find and fix bugs in your code.
- Remote Development: VS Code supports remote development, allowing you to work on projects located on remote servers, containers, or even in the Windows Subsystem for Linux (WSL).
- Integrated Git: VS Code has a built-in Git version control system that allows developers to manage source code changes efficiently.
- Extensions: The editor supports a vast ecosystem of extensions that enhance functionality, providing support for various languages, frameworks, and tools.
- Integrated Terminal: VS Code includes an integrated terminal, allowing developers to run commands directly within the editor without switching to an external terminal.
- Language Support: The editor provides rich language support for various programming languages, with features like IntelliSense (code completion), syntax highlighting, and debugging.
- Debugging: VS Code offers a powerful debugging experience with support for multiple languages and a variety of debuggers.
- Customization: Users can customize the editor's appearance and behavior, such as themes, keyboard shortcuts, and settings.
- Live Share: Collaborative coding is made easier with the Live Share extension, allowing developers to work together in real-time, even if they are in different locations.
- Containerization Support: Integration with Docker and Kubernetes for containerized development and deployment.

- Git integration: VS Code has built-in integration with Git, a popular version control system. This makes it easy to manage your code changes and collaborate with others on projects.
- Live editing: VS Code supports live editing, which allows you to see the changes you make to your code in real time. This makes it easy to experiment with different code changes and see the results immediately.

Python is a versatile and widely used high-level programming language known for its simplicity, readability, and flexibility. It was created by Guido van Rossum and first released in 1991. Python's design philosophy emphasizes code readability and ease of use, making it an excellent choice for beginners and experienced developers alike.

Key features of Python include its clean and easy-to-understand syntax, which promotes a more productive and enjoyable coding experience. The language supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Python also has a large and active community, contributing to an extensive ecosystem of libraries and frameworks that make it suitable for various applications, ranging from web development and data analysis to artificial intelligence and scientific computing.

One of Python's strengths is its cross-platform compatibility, allowing developers to write code that can run seamlessly on different operating systems. The language has gained popularity in various fields due to its versatility and the availability of numerous third-party packages. Python is frequently used in web development with frameworks like Django and Flask, in data science and machine learning with tools like NumPy and TensorFlow, and in automation and scripting tasks.

CHAPTER 3 : REQUIREMENTS & ANALYSIS

1. Problem Statement and Problem Definition

The best approach for creating an AI virtual painter will depend on the specific goals of the project. If the goal is to create paintings that are as realistic as possible, then a deep learning model is likely to be the best approach. If the goal is to create paintings that are more creative and expressive, then a rule-based system may be a better choice.

2. Requirements Specification

What is requirement analysis?

□ The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design.

a. Functional Requirements

In software engineering and systems engineering, a functional requirement defines a function of a system or its component, where a function is described as a specification of behavior between outputs and inputs.

- **Gesture Recognition**

Accurately identifying hand gestures Classifying gestures into known categories Dealing with occlusion Varying hand sizes and shapes

- **User Interface**

Developing an intuitive user interface that provides visual and auditory feedback to user in performing gestures correctly. Include clear instructions on how to perform each gesture and indicate when a gesture has been successfully recognized.

b. Non-functional Requirements

Start to write from here

In systems engineering and requirements engineering, a non-functional requirement (NFR) is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. They are contrasted with functional requirements that define specific behavior of functions.

Non-functional requirements are as follows :

- **Performance:** The AI virtual painter should be able to generate paintings in a reasonable amount of time. This will depend on the complexity of the painting and the capabilities of the hardware and software.
- **Reliability:** The AI virtual painter should be reliable and should not crash or freeze. This will depend onthe quality of the code and the robustness of the algorithms.
- **Security:** The AI virtual painter should be secure and should not be vulnerable to hacking or data breaches. This will depend on the security measures that are implemented.
- **Usability:** The AI virtual painter should be easy to use by people of all skill levels. This will depend on the user interface and the documentation.
- **Accessibility:** The AI virtual painter should be accessible to people with disabilities. This will dependon the use of accessibility features, such as screen readers and voice commands.

c. User Requirements

What is the user requirement?

User requirements for an Al virtual painter define the features, functionalities, and capabilities that users expect from the system. These requirements are typically stated in terms of what the system should do to meet the needs and expectations of its users

User requirements are as follows :

- **User-Friendly Interface:** The virtual painter should feature an intuitive and user-friendly interface that is easy for users of all skill levels to navigate.
- **Real-Time Collaboration:** Users should be able to collaborate in real-time on the same canvas, observing each other's changes as they happen.
- **Learning Resources:** The system should provide tutorials, tips, and examples of different artistic styles, aiding users in improving their painting techniques.

d. Hardware Requirements

Monitor
Camera

e. Software Requirements

A condition or capability needed by a user to solve a problem or achieve an objective. A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.

Software requirements are as follows:

- Operating system
  Windows
- RAM
  8.00 GB or 16.00 GB

3. Feasibility

A well-designed feasibility study should provide a historical background of the business or project, a description of the product or service, accounting statements, details of the operations andmanagement, marketing research and policies, financial data, legal requirements and tax obligations.Generally, feasibility studies precede technical development and project implementation.

a. Operational Feasibility

Operational feasibility for an Al virtual painter refers to the practicality and viability of implementing and using the system within a specific operational context. It assesses whether the Al virtual painter can be integrated smoothly into the existing operations and processes of an organization, taking into account technical, organizational, and resource-related factors.

b. Technical Feasibility

The technical feasibility assessment is focused on gaining an understanding of the present technical resources of the organization and their applicability to the expected needs of theproposed system.

The technical feasibility of an AI virtual painter depends on a number of factors, including the availability of data, the capabilities of the hardware and software, and the expertise of the development team

.

c. Economic Feasibility

The purpose of an economic feasibility study (EFS) is to demonstrate the net benefit of a proposed project for accepting or disbursing electronic funds/benefits, taking into consideration the benefits and costs to the agency, other state agencies, and the general public as a whole.

The economic feasibility of an AI virtual painter depends on a number of factors, including the cost of development, the cost of deployment, and the potential revenue from sales.

4. Planning and Scheduling

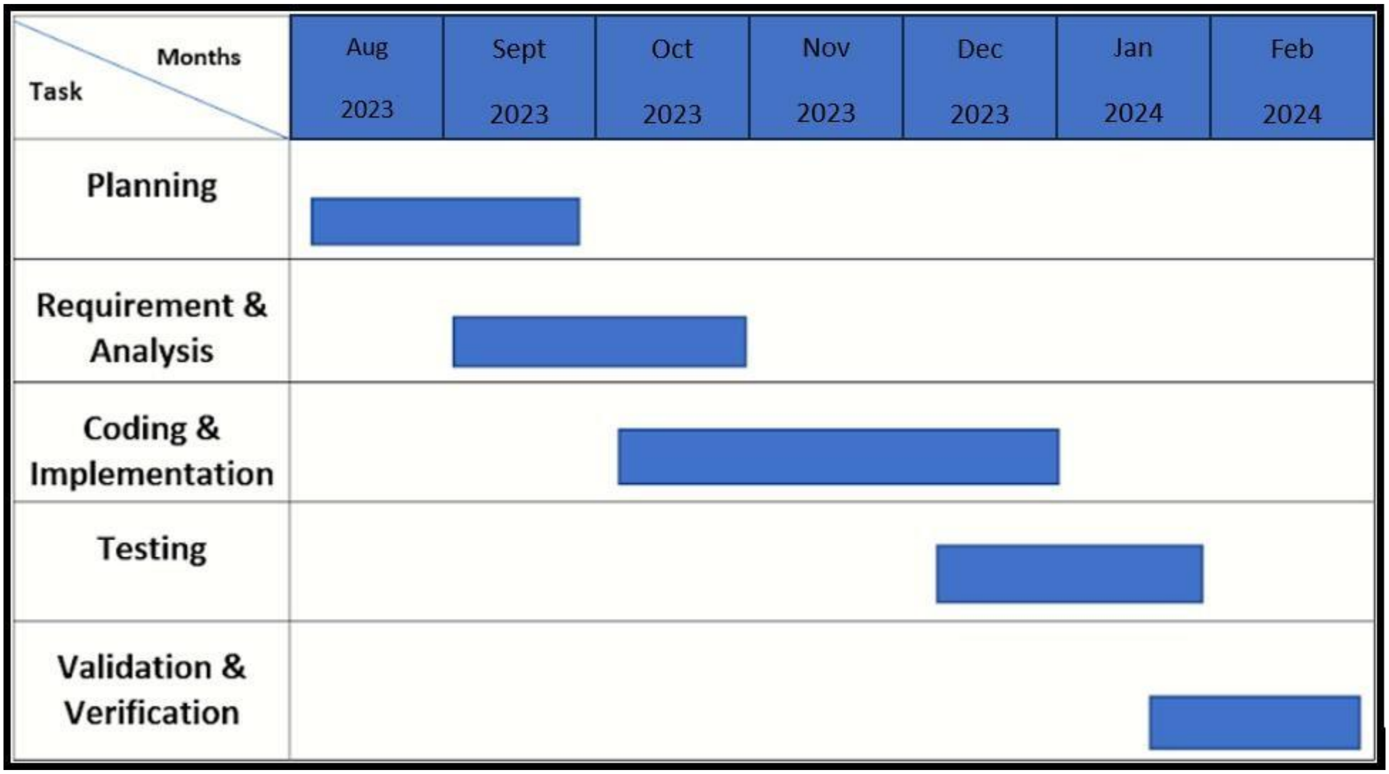
What is planning?

Planning in project management is about what steps you need to take to reach the goal, what changes and hurdle to anticipate, and how to utilize human resources and opportunities to reach the expected outcome. The planning process involves a careful analysis of the current resources and market trends and the prediction of emerging markets and future demand. Customers can easily and efficiently run the application and find the mechanics. The application should provide a reliable environment to both customers and mechanics.

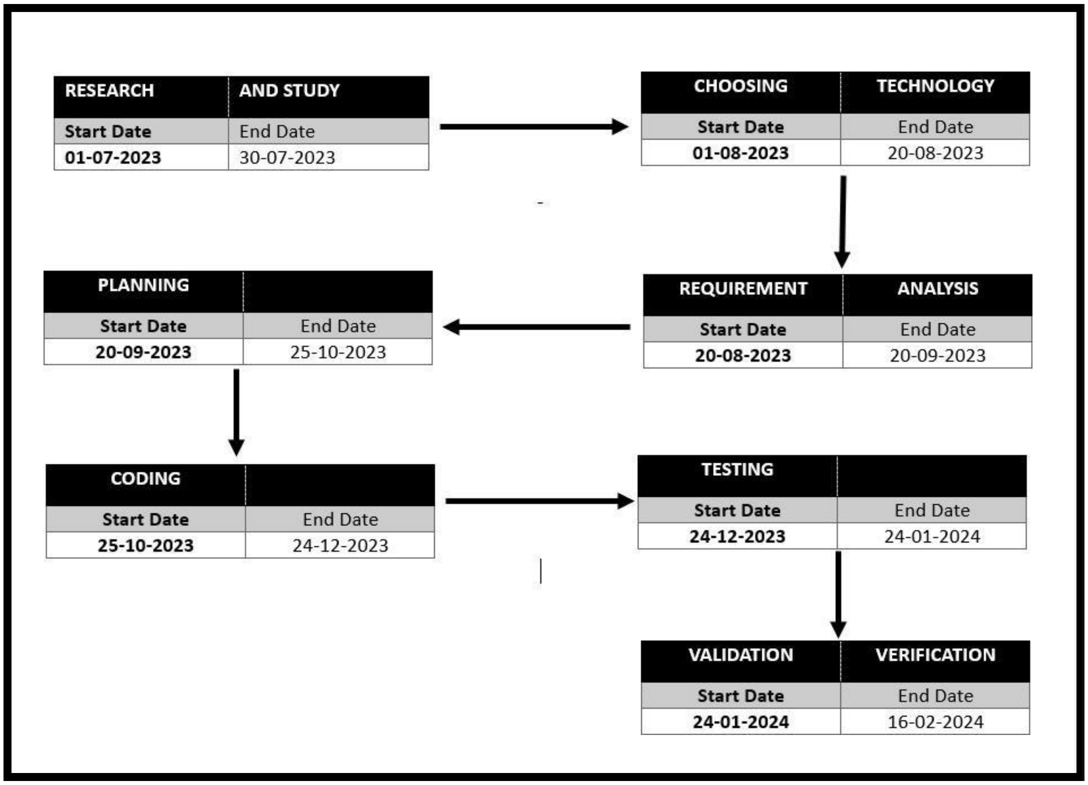
What is scheduling?

Scheduling in project management is the listing of activities, deliverables, and milestones within a project. A schedule also usually includes a planned start and finish date, duration, and resources assigned to each activity. Effective project

a. Gantt Chart



b. Pert Chart



RESEARCH	AND STUDY
Start Date	End Date
01-07-2023	30-07-2023

CHOOSING	TECHNOLOGY
Start Date	End Date
01-08-2023	20-08-2023

REQUIREMENT	ANALYSIS
Start Date	End Date
20-08-2023	20-09-2023

PLANNING	
Start Date	End Date
20-09-2023	25-10-2023

CODING	
Start Date	End Date
25-10-2023	24-12-2023

TESTING	
Start Date	End Date
24-12-2023	24-01-2024

VALIDATION	VERIFICATION
Start Date	End Date
24-01-2024	16-02-2024

5. Preliminary Product Description

Preliminary product description helps in identifying the requirements and the objectives of the new proposed product/project/system. It helps in defining the functions and associated activities or operations of the proposed product/project/system.

The AI Virtual Painter is a software application that allows users to create paintings using artificial intelligence. The application uses a large dataset of paintings to train a neural network that can generate new paintings in a variety of styles. Users can specify the desired style and subject matter of the painting, and the AI Virtual Painter will generate a painting based on their input.

The AI Virtual Painter is designed to be easy to use. Users can simply select the desired style and subject matter, and the application will generate a painting in a few seconds. The application also includes a variety of features to help users customize their paintings, such as the ability to change thecolors, the brush size, and the opacity.

The AI Virtual Painter is a powerful tool that can be used by people of all skill levels. Whether you are a beginner or an experienced artist, the AI Virtual Painter can help you create beautiful and realistic paintings.

6. Conceptual Model

a. Process Model

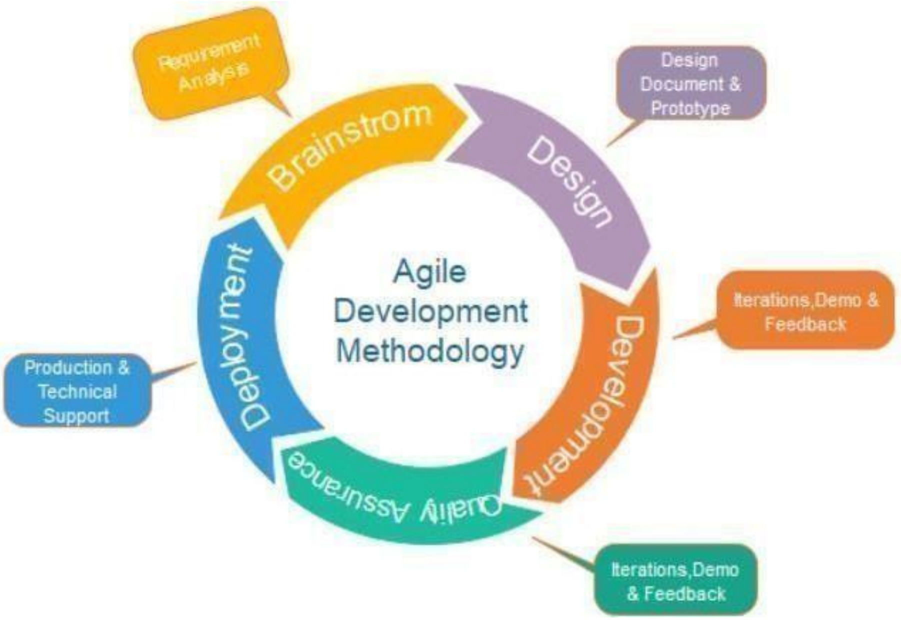
Process models are processes of the same nature that are classified together into a model. Thus, a process model is a description of a process at the type level. One possible use of a process model is to prescribe how things must/should/could be done in contrast to the process itself which is really what happens (write this as it is)

Proposed Process Model

- Agile Model

  The Agile model is a flexible and iterative approach to software development. It emphasizes collaboration, adaptability, and customer-centricity. Projects are divided into short development cycles called sprints, during which teams design, develop, test, and deliver incremental features. Regular feedback from stakeholders, including users, guides continuous refinement, allowing the project to evolve and adapt based on changing needs. The Agile model promotes

- **Design of Agile Model**


Requirement Analysis
Brainstrom
Design
Design Document & Prototype
Agile Development Methodology
Development
Iterations,Demo & Feedback
Quality Assurance
Iterations,Demo & Feedback
Deployment
Production & Technical Support

- Reasons for choosing Agile Model
    - □ Flexibility: Agile's iterative approach accommodates changing requirements and emerging insights, making it suitable for my project where specifications might evolve.
    - □ User-Centric: Agile encourages frequent user feedback, ensuring that my project remains aligned with user needs and expectations.
    - □ Faster Results: The incremental development approach yields tangible results quickly, allowing stakeholders to see progress and make informed decisions.

- Application of Agile Model
    - □ Game Development: Agile is popular in the gaming industry, where evolving gameplay mechanics and user experiences require constant adaptation and testing.
    - □ Education and Training: Agile is used in curriculum development, training programs, and e-learning to create interactive and engaging learning experiences.
    - □ Construction and Engineering: Agile practices can be adapted to manage construction projects by focusing on incremental milestones, collaborative decision-making, and adapting to changing requirements.

- Advantages of Agile Model

  - □ Customer Satisfaction: Agile's focus on regular user feedback and incremental delivery ensures that the final product aligns closely with customer expectations, resulting in higher satisfaction.
  - □ Flexibility and Adaptability: Agile embraces changing requirements and priorities, allowing teams to respond to new insights, market shifts, and evolving user needs.
  - □ Reduced Risk of Failure: Frequent reviews, testing, and user involvement catch issues early, reducing the risk of a project heading in the wrong direction for an extended period.

- Disadvantages of Agile Model
  - □ Lack of Predictability: Agile's adaptability can lead to uncertain project outcomes, making it challenging to predict exact timelines or costs.
  - □ Continuous Workload: Agile's short cycles mean a continuous workload for teams, which might lead to burnout or reduced quality if not managed properly.
  - □ Less Suitable for Large Projects: Agile might become challenging to manage in large, complex projects due to the need for frequent coordination and communication.

a. <u>The goals of a process model are to be:</u>

a. Descriptive

1. Requirement gathering - All possible requirements are captured in product requirement documents
2. Analysis – The requirements and based on analysis define the schemas, models and business rules.
3. Design – Based on analysis, design the product.
4. Implementation Development of the software in the small units with functional testing
5. Integration and testing integrating of each unit developed in previous phase and post integration test the entire system for any faults.

b. Prescriptive

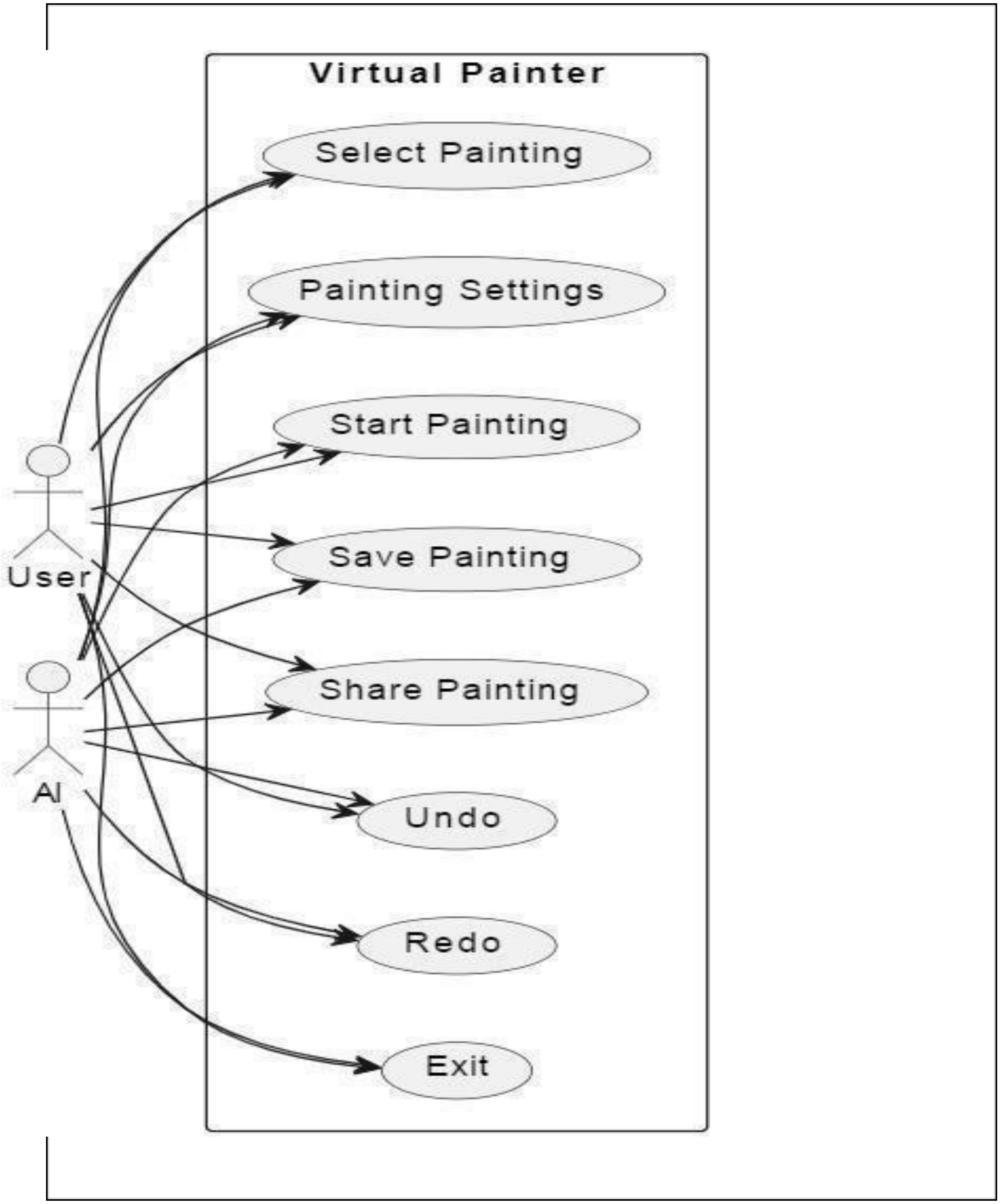
- Iterative Development: Agile development occurs in short iterations, typically referred to as "sprints." Each sprint lasts a fixed duration (e.g., 1 to 4 weeks) and results in a potentially shippable increment of the product.
- Sprint Planning: At the start of each sprint, we select a subset of user stories from the backlog to work on during the sprint. We break down the selected stories into actionable tasks.
- Sprint Execution: During the sprint, we design, develop, test, and integrate the selected user stories. This results in a potentially releasable product increment by the end of the sprint.
- Sprint Review: At the end of each sprint, a sprint review is held to showcase the completed features to stakeholders, including users. Feedback is gathered, and adjustments are made as necessary.

c. Explanatory

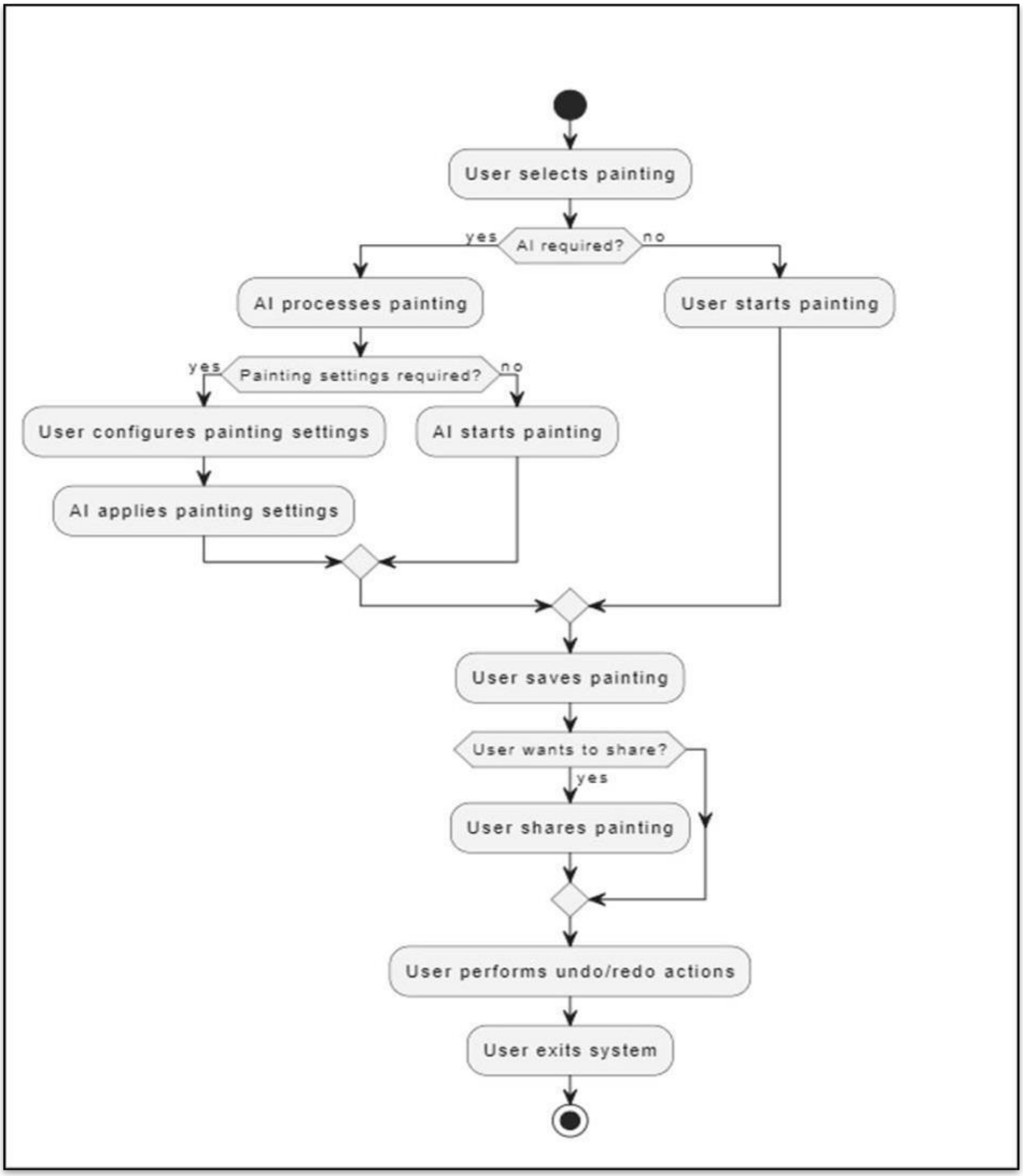
- The rationale behind these processes is to foster collaboration, deliver value incrementally, adapt to change, and ensure the final product meets user needs. Agile's processes are designed to create a dynamic and customer-centric approach to development, promoting efficiency, quality, and customer satisfaction.

c. Diagrams to be included in the design phase are as follows:

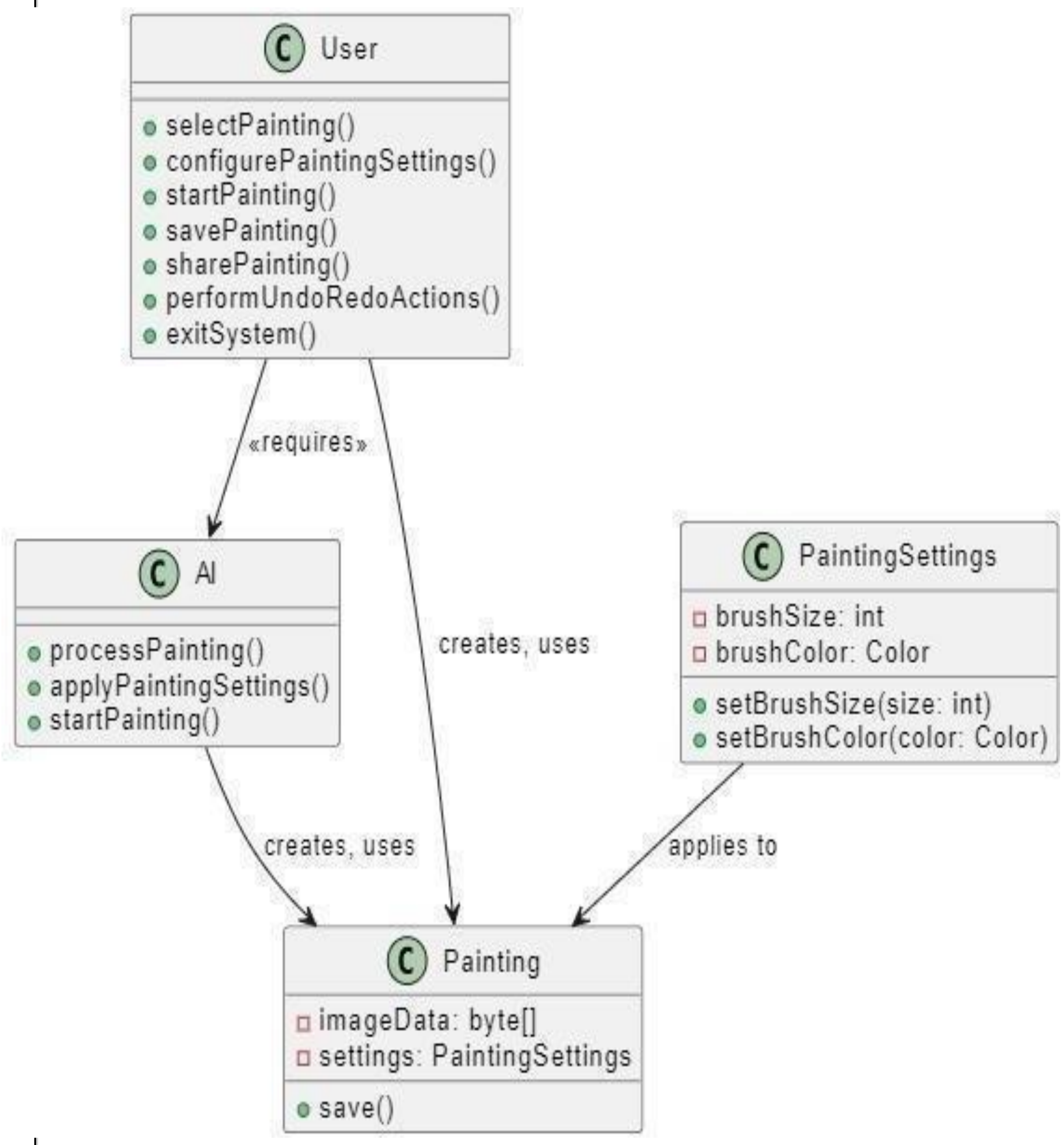
1. Use case diagram



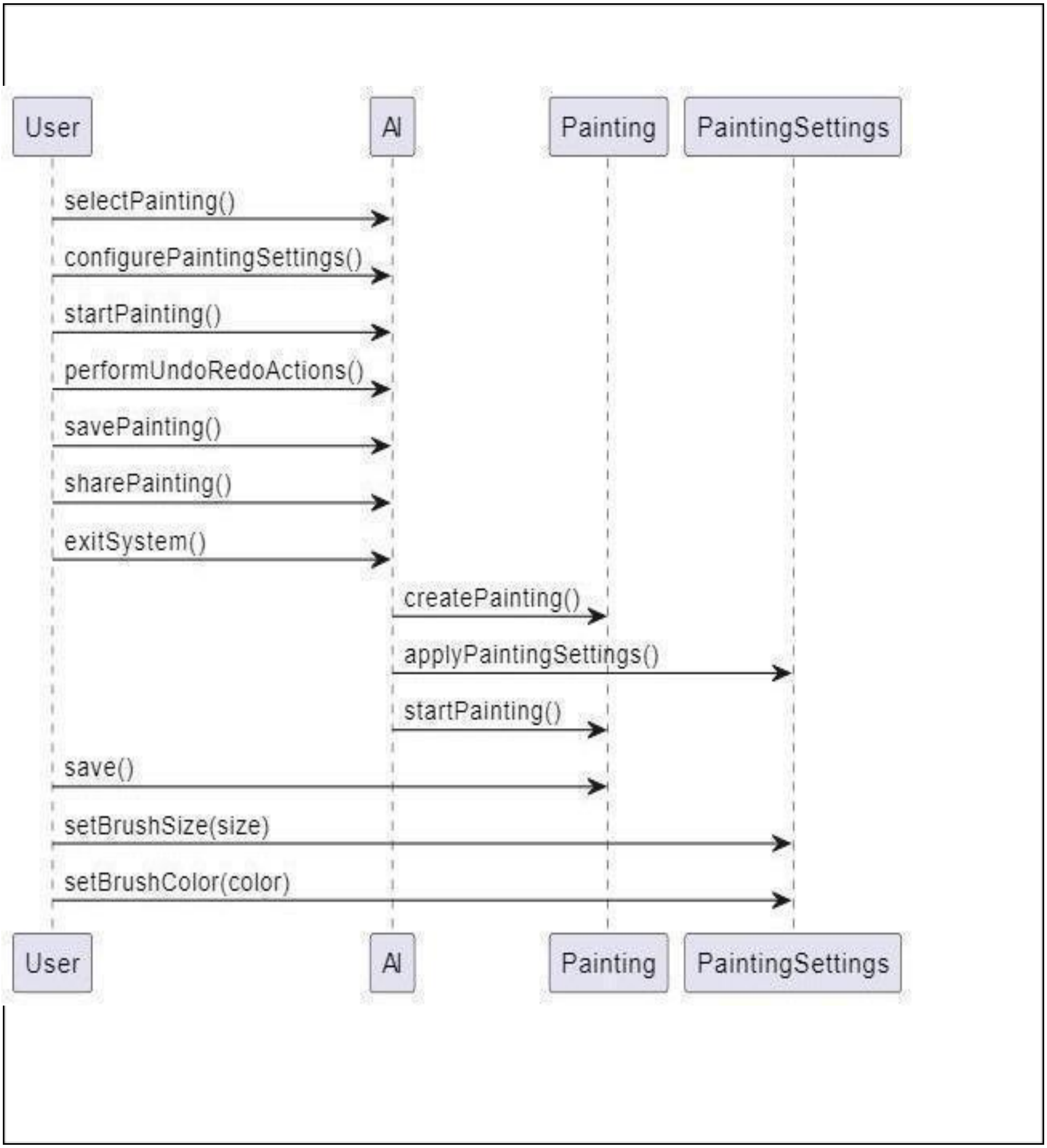
2. Activity diagram



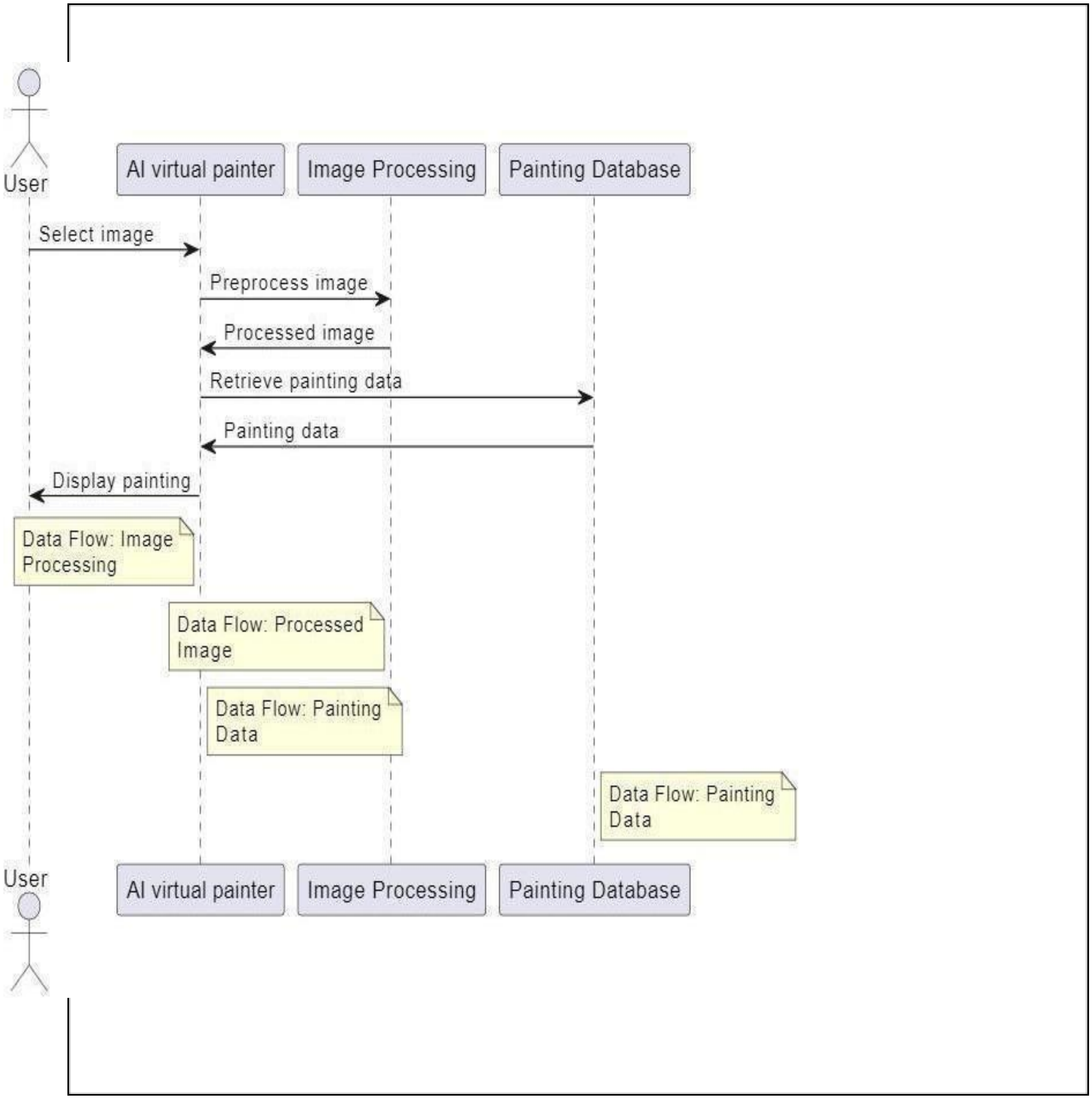
3. Class diagram



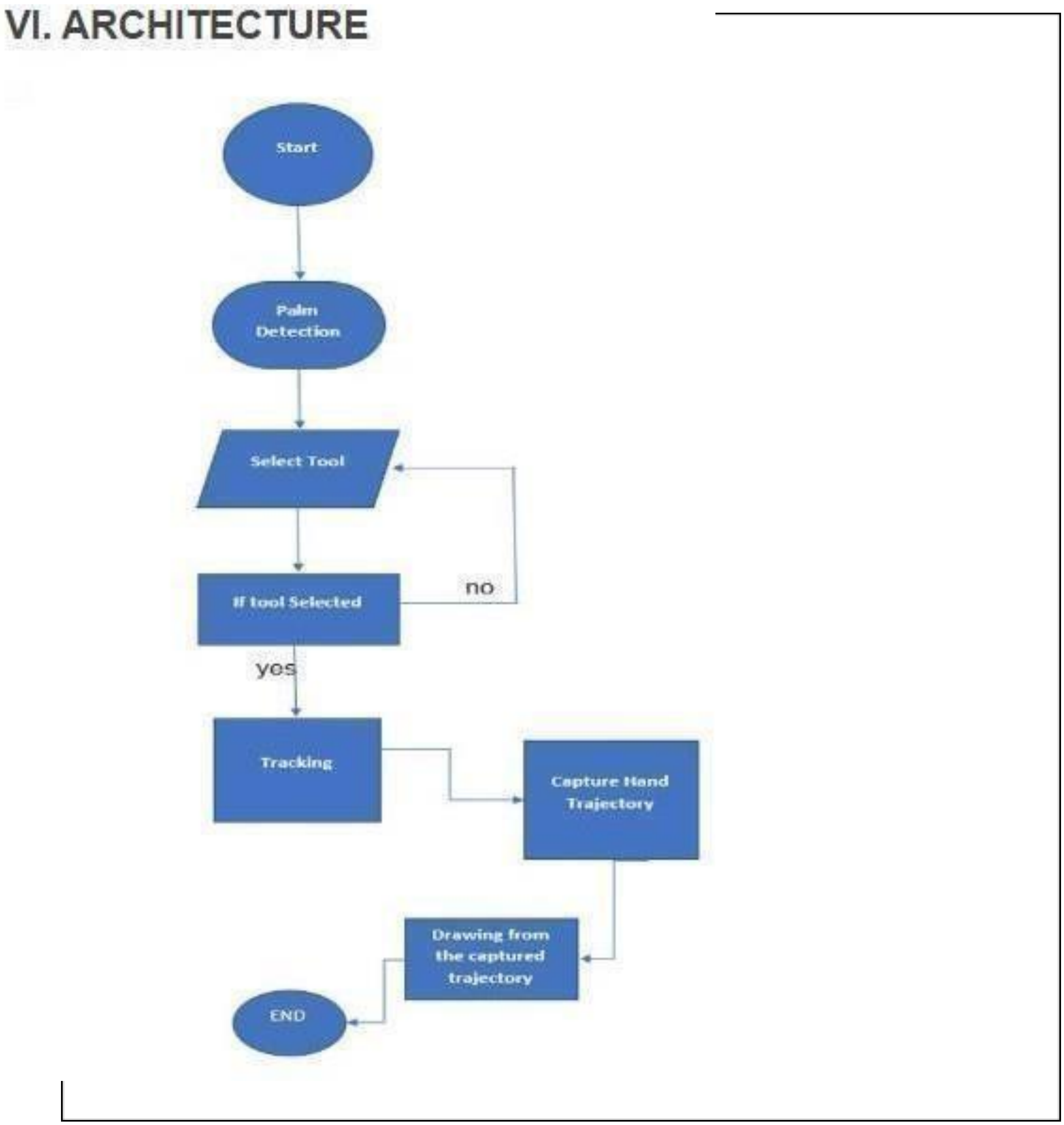
4. Sequence diagram



5. E-R model


User
AI virtual painter
Image Processing
Painting Database
Select image
Preprocess image
Processed image
Retrieve painting data
Painting data
Display painting
Data Flow: Image Processing
Data Flow: Processed Image
Data Flow: Painting Data
Data Flow: Painting Data
User
AI virtual painter
Image Processing
Painting Database

6. Data Flow Diagram


VI. ARCHITECTURE

CHAPTER 4 : SYSTEM DESIGN

1. **Basic Modules**

   a. Gesture Recognition Module

   b. Hand Tracking Module

   c. Camera

a. Description of Desired Modules

1. **Gesture Recognition Module:** This module is responsible for capturing and interpreting user gestures. It could involve using sensors or cameras to detect hand movements and translate them into an image.

2. **Hand Tracking and Persistence:** This could be used to track the user's hand movements and use them to control the painting process. For example, the user could use their hand to draw on the canvas or to select different brush sizes and colors.

b. Description of Desired Features

1. **Intuitive Gesture Control:** Users can control the cursor using natural hand gestures. Gesture recognition is accurate, responsive.

2. **Error Handling and Recovery:** Detect and handle situations where gesture recognition might fail or misinterpret gestures. Provide guidance to users on correcting gestures if needed.

2. Data Design

In the design phase, the requirements will be broken down further to be able to forecast the project’s timeline and estimate the level of effort and amount of resources needed. Design is a very important phase and is a multi-step process which represents structure, program, interface characteristics and procedural details. The proposed system is designed using the design models such as functional decomposition diagrams, data flow diagrams, entity relationship diagrams or any unified modeling language diagrams. The design phase includes all the diagrams which provide an outline of how the application would look.

3. Procedural Design

Define the desired outcome: The first step is to define the desired outcome of the procedural design process. This could be a specific painting style, a particular subject matter, or a set of desired features.

Gather data: Once the desired outcome has been defined, the next step is to gather data that will be used to create the procedural design. This data could include images, text, or code.

Create the procedural model: The next step is to create the procedural model. This is a set of instructions that will be used to generate the paintings. The procedural model can be created using a variety of programming languages and techniques.

Test the procedural model: Once the procedural model has been created, it needs to be tested. This can be done by generating a number of paintings and evaluating them against the desired outcome.

Refine the procedural model: Once the procedural model has been tested, it may need to be refined. This could involve making changes to the data, the procedural model, or the evaluation criteria.

Deploy the procedural model: Once the procedural model is satisfactory, it can be deployed. This could involve making the model available to users or incorporating it into an AI virtual painter application.

I. Algorithm design

Hand Detection Algorithm:

The cvzone library provides a pre-trained Hand Detection model, which uses a machine learning algorithm to detect and locate hands within a given frame. The algorithm uses deep learning techniques to analyze the frame and identify potential hand regions based on certain visual patterns and features

Gesture Recognition Algorithm:

The code tracks the position of the detected hands and uses the hand type , with two fingers it will select color , brushes ,eraser ; with single finger it will create image

4. User Interface Design

I. Define the user task and environments availability to carry that task

II. Describe internal and external components of the architecture or user interface

III. Draw or frame sample user interface design.

5. Security Issues

Here are some security issues that may arise with an AI virtual painter:

- Data security: The AI virtual painter will need to access a large dataset of paintings in order to train the model. This dataset could be vulnerable to hacking or data breaches, which could lead to the theft of personal information or the creation of fake paintings.
- Model security: The AI virtual painter will use a neural network to generate paintings. This neural network could be vulnerable to attacks, which could lead to the generation of malicious paintings or the corruption of the model.
- User security: The AI virtual painter will need to collect personal information from users, such as their name, email address, and payment information. This information could be vulnerable to hacking or data breaches, which could lead to identity theft or financial fraud.
- Misuse: The AI virtual painter could be used to create fake paintings or to generate paintings that are offensive or harmful. This could lead to legal problems for the user or for the developer of the AI virtual painter.

These are just some of the security issues that may arise with an AI virtual painter. It is important to carefully consider these issues before developing or using an AI virtual painter.

Here are some measures that can be taken to mitigate these security risks:

- Use strong encryption to protect data: The data used to train the AI virtual painter and the data collected from users should be encrypted to protect it from unauthorized access.
- Use secure authentication methods: Users should be authenticated using secure methods, such as two- factor authentication, to protect their accounts from unauthorized access.
- Monitor for suspicious activity: The AI virtual painter should be monitored for suspicious activity, such as attempts to generate malicious paintings or to access user data without authorization.
- Educate users about security risks: Users should be educated about the security risks associated with using an AI virtual painter and how to protect themselves.

By taking these measures, it is possible to mitigate the security risks associated with AI virtual painters.

2. Dataset - only for AI oriented projects

Name - Google Arts & Culture: Google Arts & Culture offers access to a vast collection of artworks from museums and cultural institutions worldwide.

Source - Google Images:

Use Google's image search filters to find images labeled for reuse. However, exercise caution, as not all images labeled this way are genuinely available for reuse

Size - There have been various projects and efforts by researchers and enthusiasts to curate and create datasets using images sourced from Google Images. These datasets might vary in size depending on the specific project's needs. Some researchers might collect a few thousand images, while others might gather tens of thousands or more.

Features:

- Image URL: The URL of the image's source on the web.
- Image Title/Description: The title or description associated with the image on the web page itcomes from.
- Image Source: The website or platform where the image was found.
- Image Size and Resolution: The dimensions of the image in pixels (width x height). Image Format: The file format of the image (JPEG, PNG, GIF, etc.).
- Licensing Information: If available, information about the image's licensing and usage rights.

CHAPTER 5 : IMPLEMENTATION AND TESTING

1. Implementation Approaches

Implementing an AI virtual painter involves leveraging advanced generative models Techniques such as style transfer, recurrent neural networks, and conditional generative models enhance creativity and user influence. Interactive evolutionary algorithms and deep reinforcement learning enable real-time adaptation to user preferences, while multimodal approaches and careful hyperparameter optimization contribute to the expressiveness and performance of the AI virtual painter.

a. Define the implementation plan

I started the planning for my project by firstly deciding which language would be suitable for my project. After a lot of research I finally chose Pyhton as my main coding language. Pycharm being the best platform supporting python is where I am going to implement my project “AI VIRTUAL PAINTER”.
I roughly drew the UI design on a paper and then designed it with the help of Canva. After a lot of research on similar projects I built my main code with the help of inbuilt python libraries like hand tracking module etc.

b. State the standards and protocols used in implementation

Standards and protocols in the context of AI virtual painters encompass adherence to ethical guidelines, privacy regulations, and data security measures. Compliance with industry standards such as Open Neural Network Exchange (ONNX) for model interchangeability and adherence to ethical AI principles, including transparency and fairness, are crucial. Additionally, protocols for data handling, storage, and user consent ensure responsible and secure implementation, fostering trust and accountability in the development and deployment of AI virtual painters.

2. Coding Details and Code Efficiency

The provided Python code is a script for an interactive drawing application using computer vision and the OpenCV library. It employs hand tracking to enable users to draw on a virtual canvas using their webcam. The key features include the ability to select colors by placing the hand in specific regions, draw on the canvas by moving the index finger, and dynamically change brush thickness and color. The application supports different drawing modes and includes image processing techniques to combine the drawn content with the original image. Additionally, it allows users to choose from a set of header images, and the drawing interactions are displayed in real-time on the screen.

a. Code of the main logic (must be with comments)

```
import numpy as np
import cv2
from collections import deque
import os
import time

#default called trackbar function
def setValues(x):
   print("")


# Creating the trackbars needed for adjusting the marker colour
cv2.namedWindow("Color detectors")
cv2.createTrackbar("Upper Hue", "Color detectors", 153, 180,setValues)
cv2.createTrackbar("Upper Saturation", "Color detectors", 255, 255,setValues)
cv2.createTrackbar("Upper Value", "Color detectors", 255, 255,setValues)
cv2.createTrackbar("Lower Hue", "Color detectors", 64, 180,setValues)
cv2.createTrackbar("Lower Saturation", "Color detectors", 72, 255,setValues)
cv2.createTrackbar("Lower Value", "Color detectors", 49, 255,setValues)


# Giving different arrays to handle colour points of different colour
bpoints = [deque(maxlen=1024)]
gpoints = [deque(maxlen=1024)]
rpoints = [deque(maxlen=1024)]
ypoints = [deque(maxlen=1024)]

# These indexes will be used to mark the points in particular arrays of specific colour
blue_index = 0
green_index = 0
red_index = 0
yellow_index = 0

#The kernel to be used for dilation purpose
kernel = np.ones((5,5),np.uint8)

colors = [(255, 0, 0), (0, 255, 0), (0, 0, 255), (0, 255, 255)]
colorIndex = 0

# Here is code for Canvas setup
paintWindow = np.zeros((471,636,3)) + 255
paintWindow = cv2.rectangle(paintWindow, (40,1), (140,65), (0,0,0), 2)
paintWindow = cv2.rectangle(paintWindow, (160,1), (255,65), colors[0], -1)
paintWindow = cv2.rectangle(paintWindow, (275,1), (370,65), colors[1], -1)
paintWindow = cv2.rectangle(paintWindow, (390,1), (485,65), colors[2], -1)
paintWindow = cv2.rectangle(paintWindow, (505,1), (600,65), colors[3], -1)

cv2.putText(paintWindow, "CLEAR", (49, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 0), 2, cv2.LINE_AA)
cv2.putText(paintWindow, "BLUE", (185, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv2.LINE_AA)
cv2.putText(paintWindow, "GREEN", (298, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv2.LINE_AA)
cv2.putText(paintWindow, "RED", (420, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv2.LINE_AA)
```

```
cv2.putText(paintWindow, "YELLOW", (520, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (150,150,150), 2,
cv2.LINE_AA)
cv2.namedWindow('Paint', cv2.WINDOW_AUTOSIZE)


# Loading the default webcam of PC.
cap = cv2.VideoCapture(0)

# Keep looping
while True:
    # Reading the frame from the camera
    ret, frame = cap.read()
    #Flipping the frame to see same side of yours
    frame = cv2.flip(frame, 1)
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)


    u_hue = cv2.getTrackbarPos("Upper Hue", "Color detectors")
    u_saturation = cv2.getTrackbarPos("Upper Saturation", "Color detectors")
    u_value = cv2.getTrackbarPos("Upper Value", "Color detectors")
    l_hue = cv2.getTrackbarPos("Lower Hue", "Color detectors")
    l_saturation = cv2.getTrackbarPos("Lower Saturation", "Color detectors")
    l_value = cv2.getTrackbarPos("Lower Value", "Color detectors")
    Upper_hsv = np.array([u_hue,u_saturation,u_value])
    Lower_hsv = np.array([l_hue,l_saturation,l_value])


    # Adding the colour buttons to the live frame for colour access
    frame = cv2.rectangle(frame, (40,1), (140,65), (122,122,122), -1)
    frame = cv2.rectangle(frame, (160,1), (255,65), colors[0], -1)
    frame = cv2.rectangle(frame, (275,1), (370,65), colors[1], -1)
    frame = cv2.rectangle(frame, (390,1), (485,65), colors[2], -1)
    frame = cv2.rectangle(frame, (505,1), (600,65), colors[3], -1)
    cv2.putText(frame, "CLEAR ALL", (49, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv2.LINE_AA)
    cv2.putText(frame, "BLUE", (185, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv2.LINE_AA)
    cv2.putText(frame, "GREEN", (298, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv2.LINE_AA)
    cv2.putText(frame, "RED", (420, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2, cv2.LINE_AA)
    cv2.putText(frame, "YELLOW", (520, 33), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (150,150,150), 2, cv2.LINE_AA)


  # Identifying the pointer by making its mask
    Mask = cv2.inRange(hsv, Lower_hsv, Upper_hsv)
    Mask = cv2.erode(Mask, kernel, iterations=1)
    Mask = cv2.morphologyEx(Mask, cv2.MORPH_OPEN, kernel)
    Mask = cv2.dilate(Mask, kernel, iterations=1)

    # Find contours for the pointer after idetifying it
    cnts,_ = cv2.findContours(Mask.copy(), cv2.RETR_EXTERNAL,
            cv2.CHAIN_APPROX_SIMPLE)
    center = None

    # Ifthe contours are formed
    if len(cnts) > 0:
            # sorting the contours to find biggest
    cnt = sorted(cnts, key = cv2.contourArea, reverse = True)[0]
    # Get the radius of the enclosing circle around the found contour
    ((x, y), radius) = cv2.minEnclosingCircle(cnt)
    # Draw the circle around the contour
    cv2.circle(frame, (int(x), int(y)), int(radius), (0, 255, 255), 2)
    # Calculating the center of the detected contour
    M = cv2.moments(cnt)
    center = (int(M['m10'] / M['m00']), int(M['m01'] / M['m00']))

    # Now checking if the user wants to click on any button above the screen
    if center[1] <= 65:
        if 40 <= center[0] <= 140: # Clear Button
            bpoints = [deque(maxlen=512)]
            gpoints = [deque(maxlen=512)]
            rpoints = [deque(maxlen=512)]
            ypoints = [deque(maxlen=512)]

            blue_index = 0
            green_index = 0
            red_index = 0
            yellow_index = 0

            paintWindow[67:,:,:] = 255
        elif 160 <= center[0] <= 255:
                colorIndex = 0 # Blue
        elif 275 <= center[0] <= 370:
                colorIndex = 1 # Green
        elif 390 <= center[0] <= 485:
                colorIndex = 2 # Red
        elif 505 <= center[0] <= 600:
                colorIndex = 3 # Yellow
    else :
        if colorIndex == 0:
            bpoints[blue_index].appendleft(center)
        elif colorIndex == 1:
            gpoints[green_index].appendleft(center)
        elif colorIndex == 2:
            rpoints[red_index].appendleft(center)
        elif colorIndex == 3:
            ypoints[yellow_index].appendleft(center)
# Append the next deques when nothing is detected to avois messing up
else:
    bpoints.append(deque(maxlen=512))
    blue_index += 1
    gpoints.append(deque(maxlen=512))
    green_index += 1
    rpoints.append(deque(maxlen=512))
    red_index += 1
    ypoints.append(deque(maxlen=512))
    yellow_index += 1
```

b. Code Efficiency

Effectiveness:

- User Interaction: The code effectively captures hand movements using the HandTrackingModule and translates them into drawing actions on the canvas. The ability to switch between selection and drawing modes by detecting specific finger configurations enhances user interaction.

- Drawing Features: The script supports dynamic color selection, brush thickness adjustment, and the provision for an eraser. These features contribute to a versatile drawing experience for users.

Efficiency:

- Frame Processing: The efficiency of the code depends on the speed of frame processing. The use of OpenCV functions for image manipulation and hand tracking is generally efficient, especially given the real-time requirements of webcam-based applications.

- Optimization: The implementation appears reasonably optimized, with efforts to minimize redundant computations. However, there may be opportunities for further optimization, depending on specific performance requirements.

3. Testing Approach

The testing scheme involves a multi-tiered, adapting to the specific system design. Security and user acceptance testing address vulnerabilities and user satisfaction.

a. Functional Testing

- Hand Detection and Tracking:

Objective: Verify that the system accurately detects and tracks hand movements. Test Steps:
Place the hand in the camera view and observe if the system correctly identifies and tracks hand landmarks.

- Color and Brush Selection:

Objective: Ensure that users can effectively select colors and brush thickness. Test Steps:
Switch between different color options and confirm that the drawing color changes accordingly.
Adjust the brush thickness and verify that it reflects in the subsequent drawings.

- Drawing on Canvas:

Objective: Confirm that users can draw on the virtual canvas. Test Steps:
Use hand gestures to draw lines and shapes on the canvas. Check if the drawn content is displayed in real-time.

- Mode Switching:

Objective: Verify the functionality of switching between selection and drawing modes. Test Steps:
Confirm that placing two fingers activates the selection mode.
Verify that drawing with a single finger activates the drawing mode.

1. User Acceptance Testing or Beta Testing

User Acceptance Testing (UAT) for the AI virtual painter project involves engaging end-users to evaluate the application's functionality and usability. Users will interact with the system to test features such as color and brush selection, drawing on the canvas, mode switching, header image selection, canvas clearing, real- time display, image processing, user interaction, and error handling. The testing aims to ensure that the application meets user requirements, provides an intuitive interface, and performs effectively in real-world scenarios. User feedback will be collected to identify any issues, improvements, or suggestions, and the testing process will be iterative, with updates and refinements made based on user input to ensure a positive and user-friendly experience.

2. Unit Testing

Unit testing for the AI virtual painter project involves systematically examining individual components or functions of the code to ensure they perform as intended. Test cases will be designed to verify the correctness of critical functions, such as hand detection and tracking, color and brush selection, drawing on the canvas, mode switching, and image processing. These tests aim to identify any bugs, logical errors, or unexpected behavior within isolated parts of the codebase. By validating each unit's functionality independently, developers can ensure the reliability and stability of the overall system. Unit testing is an essential practice for maintaining code quality, facilitating early bug detection, and supporting codebase maintainability throughout the development lifecycle.

3. Integration Testing

Integration testing for the AI virtual painter project involves assessing the interactions and collaborations between different modules, components, and external dependencies to ensure seamless integration and proper functioning of the overall system. Test cases will focus on verifying the communication and data flow between key elements, such as hand tracking, color and brush selection, drawing features, and image processing. This testing phase aims to identify any issues arising from the integration of these components, ensuring that they work harmoniously to deliver the intended functionality. By detecting and addressing integration issues early in the development process, integration testing contributes to the creation of a robust and cohesive AI virtual painter system, minimizing the risk of errors or inconsistencies during runtime.

b. Non-Functional Testing

Non-functional testing for the AI virtual painter project involves evaluating aspects of the system that go beyond its specific functionalities, focusing on performance, usability, reliability, and security. Performance testing assesses the application's responsiveness and scalability under varying conditions, ensuring it meets expected speed and resource requirements. Usability testing examines the user interface and overall user experience, ensuring that the application is intuitive and user-friendly. Reliability testing verifies the system's stability and availability over extended periods, while security testing assesses vulnerabilities and safeguards against potential threats. By conducting comprehensive non-functional testing.

1. Performance Testing

Performance testing for the AI virtual painter project involves assessing the application's responsiveness, scalability, and resource utilization under different conditions to ensure optimal functionality. The testing will focus on evaluating the speed and efficiency of key features such as hand tracking, drawing interactions, and image processing. Load testing will examine how the system performs under various user loads, ensuring that it remains responsive and stable during peak usage. Additionally, stress testing will assess the application's robustness by pushing it to its limits, verifying that it can handle high-demand scenarios without compromising performance. The objective is to identify and address any bottlenecks, latency issues, or resource constraints, ensuring that the AI virtual painter meets performance requirements and provides a seamless and reliable user experience.

2. Scalability Testing

Scalability testing for the AI virtual painter project involves evaluating the application's ability to handle increased workloads and user demands gracefully. The testing process will assess how well the system scales with growing datasets, diverse artistic inputs, and concurrent user interactions. By simulating scenarios where the application experiences increased usage or higher data volumes, scalability testing aims to identify potential limitations and optimize the system's performance. The objective is to ensure that the AI virtual painter can efficiently scale to accommodate a growing user base and increasing complexity while maintaining a consistent level of responsiveness and functionality. Addressing scalability concerns proactively will contribute to the project's ability to meet future demands and ensure a smooth user experience as the application scales.

3. Portability Testing

Portability testing for the AI virtual painter project involves assessing the application's ability to function seamlessly across diverse environments, devices, and operating systems. This testing phase ensures that the project can be easily transferred or adapted to different platforms without compromising its core functionality. Test cases will focus on verifying compatibility with various web browsers, operating systems, and screen resolutions. The goal is to identify and address any issues related to dependencies, device-specific constraints, or environmental variations that could impact the application's performance or user experience. Successful portability testing will guarantee that the AI virtual painter remains accessible and functional across a wide range of devices and platforms, enhancing its usability and reach.

4. Black Box Testing

Black box testing for the AI virtual painter project involves assessing the application's functionality without knowledge of its internal code or implementation details. Test cases are designed based on the system's specifications and expected behaviors, focusing on inputs and expected outputs. The testing process encompasses various scenarios, such as different hand movements for drawing, color and brush selections, and header image choices, while evaluating the overall system response. The goal of black box testing is to validate that the application meets its specified requirements, responds appropriately to user interactions, and produces the expected outputs. By treating the system as a "black box," this testing approach ensures that the AI virtual painter

5. White Box Testing

White box testing for the AI virtual painter project involves a detailed examination of the internal structure, logic, and code of the application. Test cases are designed based on an understanding of the source code, aiming to validate individual functions, control flows, and code paths. This testing approach assesses the correctness of hand tracking algorithms, color and brush selection mechanisms, drawing features, and other internal components. By scrutinizing the code at a granular level, white box testing helps uncover issues such as logical errors, boundary cases, and potential performance bottlenecks. The primary objective is to ensure that each component functions as intended and that the overall codebase adheres to coding standards, enhancing the reliability and maintainability of the AI virtual painter.

1. **Test Cases**

TEST CASE							
System Name :	AI Virtual Painter						
Module Code :	FM001 - Importing Images						
Pass	2	Pending	0				
Fail	0	Number of test cases:	2				
ID	**Test Case Description**	**Test Case Procedure**	**Expected Output**	**Acutal Output**	**Test date**	**Result**	**Note**
Test Case 1	Making the UI design for the user to select different colours of brush	1. Making the UI design in canva using different templates 2. Saving it on my desktop and importing in the code	Paint brushes with different colours should be visible when the project is in running state	We can see Paint brushes with different colours when the project is in running state	12th October 2023	Pass	
Test Case 2	Making the UI design for the user to select Eraser	1. Making the UI design in Canva using different elements 2. Saving it on my desktop and importing in the code	Eraser should be visible when the project is in running state	We can see Eraser when the project is in running state	12th October 2023	Pass	

TEST CASE			
System Name:	AI Virtual Painter		
Module Code:	FM002 - Hand Tracking		
Pass	3	Pending	0
Fail	0	Number of test cases:	3

ID	Test Case Description	Test Case Procedure	Expected Output	Acutal Output	Test date	Result	Note
Test Case 1	Using Hand Tracking module to get accurate gestures	Importing the Hand Tracking module in Pycharm	When we will run this module ,we shall se our hands gestures getting recognized through the camera	We can see our hand gestures getting recognized	20 October 2023	Pass	
Test Case 2	Tip of index and middle fingers	1. Import hand tracking module 2. Giving coordinates to recognize index and middle fingers	We shall see the paint brushes selected	We can see the paint brushes selected	20 October 2023	Pass	
Test Case 3	Check which fingers are up	1. Importing the Hand Tracking module in Pycharm 2. Using detector.fingersUp()	We shall see which fingers are up	We can see which fingers are up	25 October 2023	Pass	

TEST CASE			
System Name :	AI Virtual Painter		
Module Code :	FM003 - Drawing Mode		
Pass	1	Pending	0
Fail	1	Number of test cases:	2

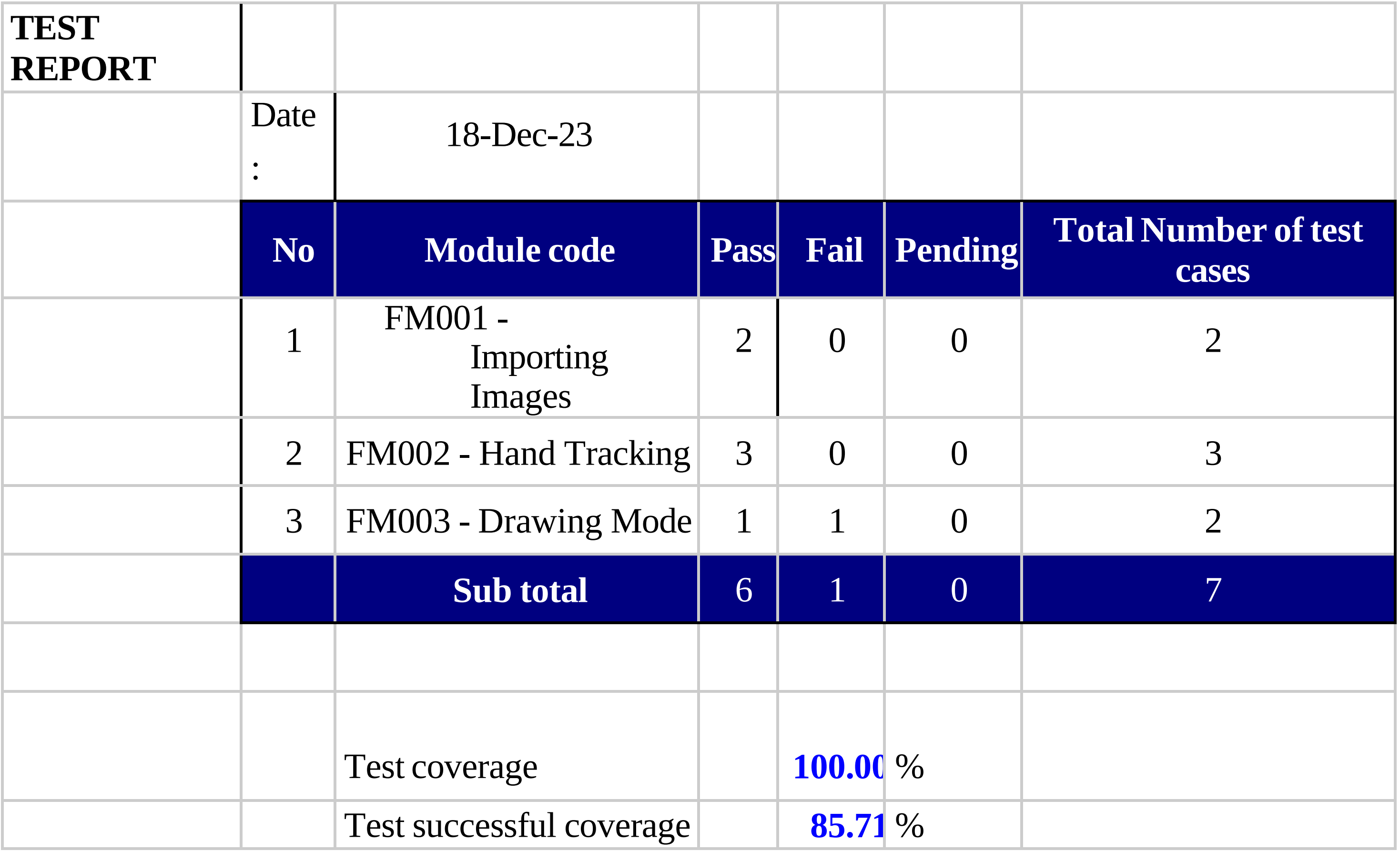
ID	**Test Case Description**	**Test Case Procedure**	**Expected Output**	**Acutal Output**	**Test Date**	**Result**	**Note**
Test Case 1	Enable drawing mode if finger is up	1. With the help of hand tracking module checking if the finger is up 2. Using the finger function 3. Giving [1] as parameter for function fingers	We should be able to draw with the selected brush	We can draw with the selected brush	30 October 2023	Pass	
Test Case 2	Autocorrecting spelling mistakes	1. Importing dictionary module 2. Getting the closest match to the words	Spelling mistakes should be auto corrected	We cannot see spelling mistakes getting autocorrected	31 October 2023	Fail	

d. Modification and Expected Improvements

For the AI virtual painter project, focus on enhancing user experience and artistic capabilities. Implement a more intuitive and feature-rich user interface, offering advanced brush and stroke techniques for increased creativity. Integrate machine learning models to better understand and replicate diverse art styles, and explore style transfer techniques for added versatility. Provide real-time feedback, personalized customization options, and collaborative painting features to engage users and foster a sense of community. Optimize performance, introduce gamification elements, and facilitate easy sharing of artworks through various export options and social media integration. These modifications aim to create a more immersive and enjoyable virtual painting experience, appealing to both novice and experienced artists alike.

CHAPTER 6 : RESULTS AND DISCUSSIONS

1. Test Reports



TEST REPORT						
	Date :	18-Dec-23				
	No	**Module code**	**Pass**	**Fail**	**Pending**	**Total Number of test cases**
	1	FM001 - Importing Images	2	0	0	2
	2	FM002 - Hand Tracking	3	0	0	3
	3	FM003 - Drawing Mode	1	1	0	2
		Sub total	6	1	0	7
		Test coverage		**100.00**	%	
		Test successful coverage		**85.71**	%	

- AI Virtual Painter
- Test Objective
  - Write objective of each stage of testing i.e. testing for each module and also functional, non functional testing
- Test Summary
  - Take this from excel sheet but frame it in the form of sentences

2. Cost Estimation

Cost estimation models are mathematical algorithms or parametric equations used to estimate the costs of a product or project. The results of the models are typically necessary to obtain approval to proceed, and are factored into business plans, budgets, and other financial planning and tracking mechanisms.

- **The Development Model**

COCOMO (Constructive Cost Model) is a regression model based on LOC viz. number of Lines of Code. It is a procedural cost estimate model for software projects and often used as a process of reliably predicting the various parameters associated with making a project such as size, effort, cost, time and quality.

- **Key Parameter**

a. Efforts - measured in person months units
b. Schedule - measured in span of months or weeks

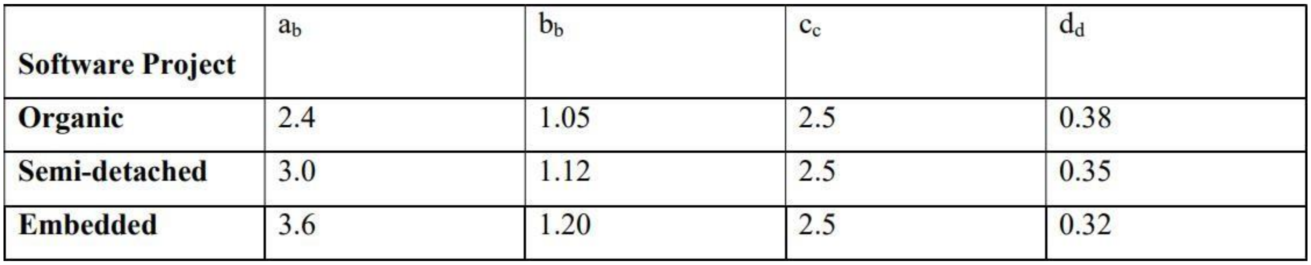
To estimate the effort and development time, COCOMO uses the same equations but with different coefficients (a, b, c, d in the effort and schedule equations) for each development mode. Types are as follows :

- Organic System
- Semi - detached System
- Embedded System

The basic COCOMO equations take the form

- Effort Applied (E) = ab (KLOC) bb [person-months]
- Development Time (D) = cb (Effort Applied) db [months]
- People Required (P) = Effort Applied / Development time [count]

Where, KLOC is the estimated number of delivered lines (expressed in thousands) of code for a project. The coefficient ab, bb, cb and db are given in the following table:



Software Project	a_b	b_b	c_c	d_d
Organic	2.4	1.05	2.5	0.38
Semi-detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

<u>COCOMO Model for "AI VIRTUAL PAINTER"</u>

a. Effort : 2.4 * 15KLOC $^{1.05}$ = 41.22 person-months

b. Time for development : 2.5 * 41.22 $^{0.38}$ = 11.76
months Where,
Effort = Number of staff months
(SM) Size = Number of source lines
of code
Time = Total number of months required to complete the project

The Project Code for AI Virtual Painter application contains 0000 Lines of

code Since, we know that 1000 Lines of Code = 1 KLOC (K - Kilo - 10^3)
Therefore, the project consists of 15000 KLOC.

Effort = = 42.21 SM

Time for development = = 11.76 Months Cost per Month = Rs.818/-
Total Cost of the Project = Cost per Month * Time required for the development project
= 818 * 11.76
= Rs. 9620 approx

CHAPTER 7 : CONCLUSIONS

1. Conclusion

The development of AI virtual painters represents a remarkable intersection of technology and creativity, offering exciting possibilities while also presenting a set of challenges and limitations. The current state of AI virtual painting showcases impressive capabilities in generating visually appealing artworks, yet it is crucial to acknowledge the constraints that persist.

AI virtual painters excel in replicating established styles and patterns learned from extensive training datasets. However, their creative output often lacks the depth of originality and conceptualization seen in human-generated art. The challenge lies in pushing the boundaries of AI to not only mimic existing artistic expressions but to transcend them and create genuinely innovative works.

Understanding abstract concepts, adapting to real-time changes, and grasping the subjective nuances of art remain areas where AI virtual painters face hurdles. Additionally, ethical considerations surrounding authorship, intellectual property, and the potential biases embedded in training data raise important questions about the role of AI in the creative process.

As technology continues to advance, addressing these limitations becomes pivotal for the evolution of AI virtual painters. Researchers and developers are actively working on refining algorithms, enhancing contextual understanding, and broadening the range of styles and concepts that AI can grasp. The future promises exciting possibilities for AI virtual painters to become more adaptive, creative, and capable of collaborating seamlessly with human artists.

In undertaking this project, it is essential to recognize the potential of AI virtual painters as powerful tools for inspiration, augmentation, and collaboration in the realm of art. While challenges persist, the journey toward overcoming these limitations contributes to the ongoing dialogue about the intersection of artificial intelligence and creativity, pushing the boundaries of what is achievable in the fascinating world of digital art.the development of AI virtual painters represents a remarkable intersection of technology and creativity, offering exciting possibilities while also presenting a set of challenges and limitations. The current state of AI virtual painting showcases impressive capabilities in generating visually appealing artworks, yet it is crucial to acknowledge the constraints that persist.

AI virtual painters excel in replicating established styles and patterns learned from extensive training datasets. However, their creative output often lacks the depth of originality and conceptualization seen in human-generated art. The challenge lies in pushing the boundaries of AI to not only mimic existing artistic expressions but to transcend them and create genuinely innovative works.

Understanding abstract concepts, adapting to real-time changes, and grasping the subjective nuances of art remain areas where AI virtual painters face hurdles. Additionally, ethical considerations surrounding authorship, intellectual property, and the potential biases embedded in training data raise important questions about the role of AI in the creative process.

As technology continues to advance, addressing these limitations becomes pivotal for the evolution of AI virtual painters. Researchers and developers are actively working on refining algorithms, enhancing contextual understanding, and broadening the range of styles and concepts that AI can grasp. The future promises exciting possibilities for AI virtual painters to become more adaptive, creative, and capable of collaborating seamlessly with human artists.

In undertaking this project, it is essential to recognize the potential of AI virtual painters as powerful tools for inspiration, augmentation, and collaboration in the realm of art. While challenges persist, the journey toward overcoming these limitations contributes to the ongoing dialogue about the intersection of artificial intelligence and creativity, pushing the boundaries of what is achievable in the
fascinating world of digital art.

2. Limitations

Limitation 1 : **Technical limitations**

Description: AI virtual painters are still under development, and they are often limited by technical constraints. For example, they may struggle to generate high-resolution images or to produce art that is realistic in appearance.

Limitation 2 **: Lack of control**

Description: AI virtual painters are not always easy to control. It can be difficult to get them to produce the exact type of art that you want. Additionally, they may make mistakes that are difficult to correct.

Limitation 3 : **Integration Challenges**

Description**:** Implementing AI virtual painters into existing creative workflows or software can be challenging due to compatibility issues and differing user interfaces.

3. Future Scope of the Project

The future of the AI virtual painter project holds great promise as advancements in artificial intelligence continue to unfold. One key avenue for development lies in enhancing the creative capabilities of the virtual painter. While the current iteration demonstrates proficiency in mimicking existing artistic styles, future iterations can strive towards achieving a higher level of originality and creativity. Research efforts may focus on developing more sophisticated generative models, incorporating advanced creative algorithms, and exploring ways to imbue the AI with a deeper understanding of abstract concepts and emotions in art.

In terms of scalability and adaptability, there is potential to expand the project's scope to accommodate a broader range of artistic genres, techniques, and cultural influences. By incorporating a more diverse and extensive training dataset, the virtual painter can become more versatile, producing artworks that reflect a richer spectrum of artistic expressions. Additionally, exploring the integration of multimedia elements, such as combining visual and auditory components, could contribute to a more immersive and multisensory artistic experience.

Areas Not Included but Potentially Addressed Later:

While the current project focuses on the generation of 2D visual artworks, future iterations could explore three-dimensional art forms. This expansion into the realm of 3D art creation would involve additional complexities but could open new possibilities for virtual sculpting, immersive environments, and interactive installations. Incorporating user-driven narrative elements into the generated art could further enhance the project, allowing users to influence the storytelling aspect of the artworks.

The project could also benefit from incorporating a feedback loop mechanism to learn and adapt from user interactions in real-time. This iterative learning process would enable the AI virtual painter to evolve and improve its creative output based on user preferences and feedback, fostering a more collaborative and dynamic artistic experience.

Extension through Research and Funding:

To extend the project through further research and funding, exploration into cutting-edge AI techniques, such as reinforcement learning and transformer architectures, could be pursued. Investing in more extensive and diverse training datasets, potentially curated in collaboration with artists from various cultural backgrounds, would contribute to a more inclusive and globally resonant virtual painter.

Moreover, securing funding for interdisciplinary collaborations with cognitive scientists and psychologists could facilitate the integration of a deeper emotional intelligence into the AI virtual painter. Understanding the psychological aspects of art creation and appreciation could inform the development of AI algorithms that not only generate visually stunning pieces but also evoke specific emotional responses.

Factors for Modification and Future Research:

Several factors within the current project can be modified and expanded for future research. Firstly, refining the algorithm to better understand and interpret abstract concepts, emotions, and subjective artistic expressions would be a key focus. Additionally, investigating ways to reduce biases present in the training data and addressing ethical considerations related to intellectual property and cultural representation are vital aspects of future research.

REFERENCES

Hanumanth Raju R K, Archana J R," **AI Virtual Printer** " ,Volume & Issue : ICEI – 2022 (Volume 10 – Issue 11) , |30-08-2022
[2] Sandesh Khore , Prof. Sunil Rathod," **AI Virtual Printer** ", Volume 10 Issue |May 2022.

https://developers.google.com/mediapipe/solutions/vision/gesture_recognizer/python http://www.diva-portal.org/smash/get/diva2:519237/FULLTEXT01.pdf
https://core.ac.uk/download/pdf/53188106.pd

"OpenGL SuperBible: Comprehensive Tutorial and Reference" by Graham Sellers, Richard S. Wright, Nicholas Haemel: This book covers OpenGL programming in detail, which can be useful for implementing virtual painting features in Python using libraries like PyOpenGL.

"Pygame Documentation": Pygame is a set of Python modules designed for writing video games. It can be utilized for implementing virtual painting interfaces in Python. The official Pygame documentation provides tutorials and references to get started.

"Learning Python" by Mark Lutz: Since you're using Python, having a solid understanding of the language itself is crucial. This book is a comprehensive guide to Python programming, covering everything from basic syntax to more advanced topics.

Online Resources and Tutorials: Websites like Real Python, Pygame tutorials on YouTube, and OpenGL tutorials for Python can offer step-by-step guides and code examples for implementing virtual painting features using Python and VSCode.