IMAGE GENERATION

WITH

STYLE TRANSFER

ABSTRACT

- **In the world of digital art, Image Style Transfer is a captivating technique that allows artists and enthusiasts to infuse their pictures with the charm of famous artistic styles.**

- **This process involves seamlessly applying the visual elements of one image onto another, resulting in a harmonious blend of content and style.**

- **This abstract explores the natural and intuitive aspects of Image Style Transfer, shedding light on how enthusiasts can effortlessly transform their photographs into visually striking compositions**

KEY WORDS:

- **Neural Style Transfer (NST),Texture Synthesis, Optimization Methods, Instance Normalization**

REQUIREMENT ANALYSIS

➢ FUNCTIONAL REQUIREMENTS

- ✓ Uploading content and style images.
- ✓ Implementing the style transfer algorithm.
- ✓ Displaying the stylized output to the user.
- ✓ Providing options for users to adjust parameters or select styles.

➢ USER REQUIREMENTS

- ❑ No need of any heavy requirements.

- ✓ Needed to upload a content image and style image

➢ NON FUNCTIONAL REQUIREMENTS

- These are qualities or attributes the system must have, but they don't relate directly to specific behaviors.

Attributes: Performance, Scalalibity, Usability, Security

➢ SYSTEM REQUIREMENTS

- Detail the hardware, software, and network requirements.
- For example:
  - **Hardware:** Specify the minimum and recommended hardware specifications for running the application.
  - **Software:** Specify the required software dependencies, frameworks, and libraries.

MODULES DESCRIPTION

1.TensorFlow (or Py Torch):

1. **Description:** Deep learning frameworks that provide tools and abstractions for building and training neural networks.
2. **Use:** Define and train the style transfer model. TensorFlow and PyTorch offer high-level APIs that simplify the implementation.

2.NumPy:

1. **Description:** A library for numerical operations in Python.
2. **Use:** Manipulate and process arrays and matrices, which are fundamental for image data handling.

3.OpenCV:

1. **Description:** A computer vision library with tools for image and video processing.
2. **Use:** Read, manipulate, and display images. Useful for preprocessing and post-processing steps.

4.PIL (Pillow):

1. **Description:** Python Imaging Library (Pillow) is a library for opening, manipulating, and saving various image file formats.
2. **Use:** Handle image-related tasks such as loading, saving, and basic transformations.

5.Matplotlib:

1. **Description:** A 2D plotting library for Python.
2. **Use:** Visualize images, plots, and other graphical representations during the development process.

6.Jupyter Notebooks:

1. **Description:** An interactive computing environment.
2. **Use:** Develop and document code in an interactive and visual manner. Useful for experimenting with different parameters.

FEASIBILITY STUDY

1. Project Scope and Objectives

- **Scope:** Implementing a style transfer algorithm for generating artistic images.
- **Objectives:** Create a user-friendly application for transforming content images with artistic styles.

2. Technical Feasibility

- **Algorithm Selection:**
  - Investigate feasibility of implementing style transfer algorithm.
  - Assess computational requirements and available libraries/tools.
- **Data Requirements:**
  - Evaluate availability and quality of datasets for training/testing.

3. Market Feasibility

- **Identify Users:**
  - Define target audience and understand their needs.
- **Competitive Analysis:**
  - Analyze existing solutions and competitors in style transfer space.

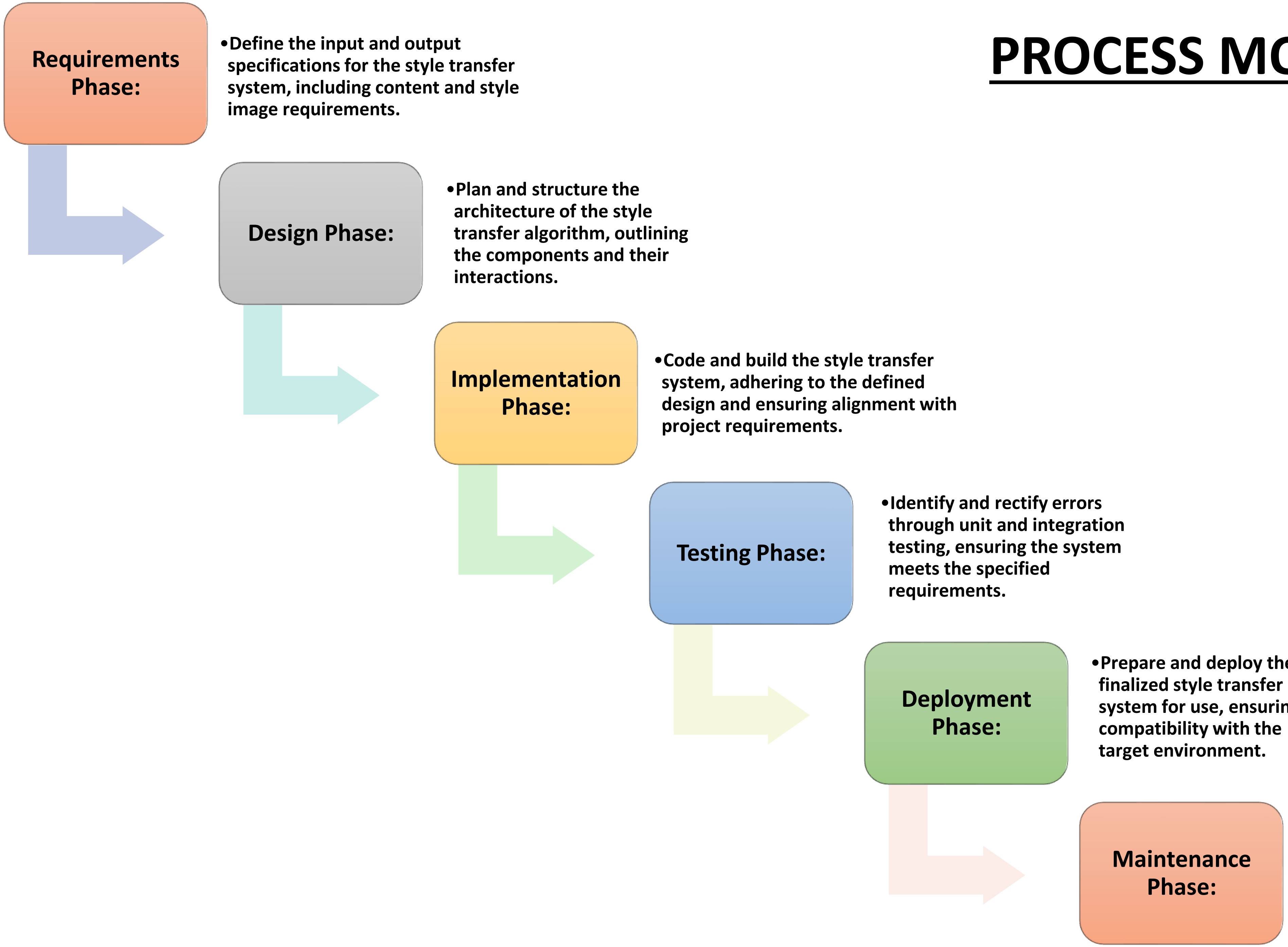
4. Financial Feasibility

- **Cost Estimates:**
  - Estimate development, training, testing, and deployment costs.
- **Revenue Model:**
  - Explore potential revenue sources or benefits.

5. Operational Feasibility

- **Resource Availability:**
  - Ensure necessary skills, tools, and infrastructure are available/acquired.
- **Operational Processes:**
  - Outline steps involved in system operation.

PROCESS MODEL



Requirements Phase:

- Define the input and output specifications for the style transfer system, including content and style image requirements.

Design Phase:

- Plan and structure the architecture of the style transfer algorithm, outlining the components and their interactions.

Implementation Phase:

- Code and build the style transfer system, adhering to the defined design and ensuring alignment with project requirements.

Testing Phase:

- Identify and rectify errors through unit and integration testing, ensuring the system meets the specified requirements.

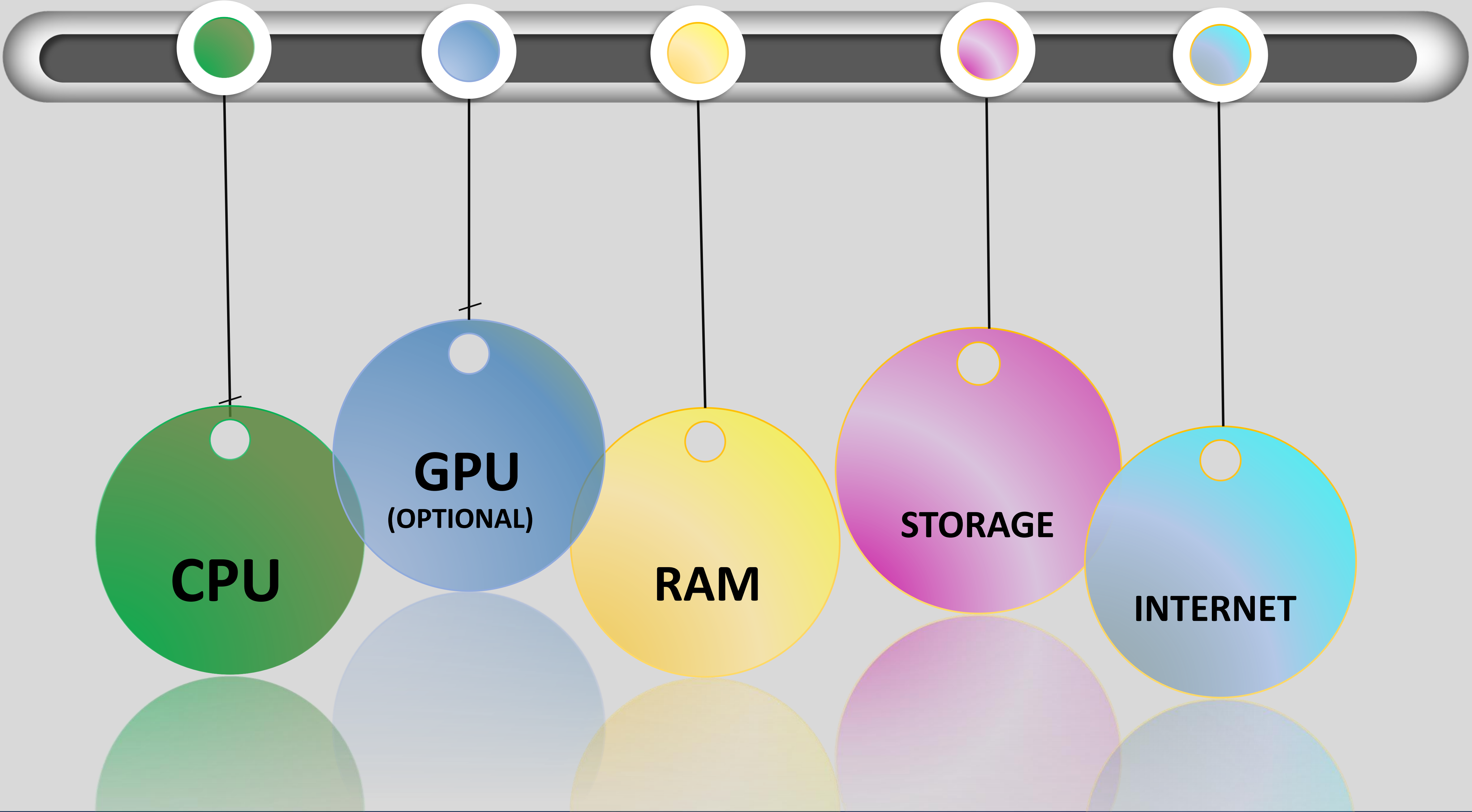
Deployment Phase:

- Prepare and deploy the finalized style transfer system for use, ensuring compatibility with the target environment.

Maintenance Phase:

- Monitor the system, address reported issues, and make updates or improvements based on user feedback and evolving needs.

HARDWARE REQUIREMENTS



CPU

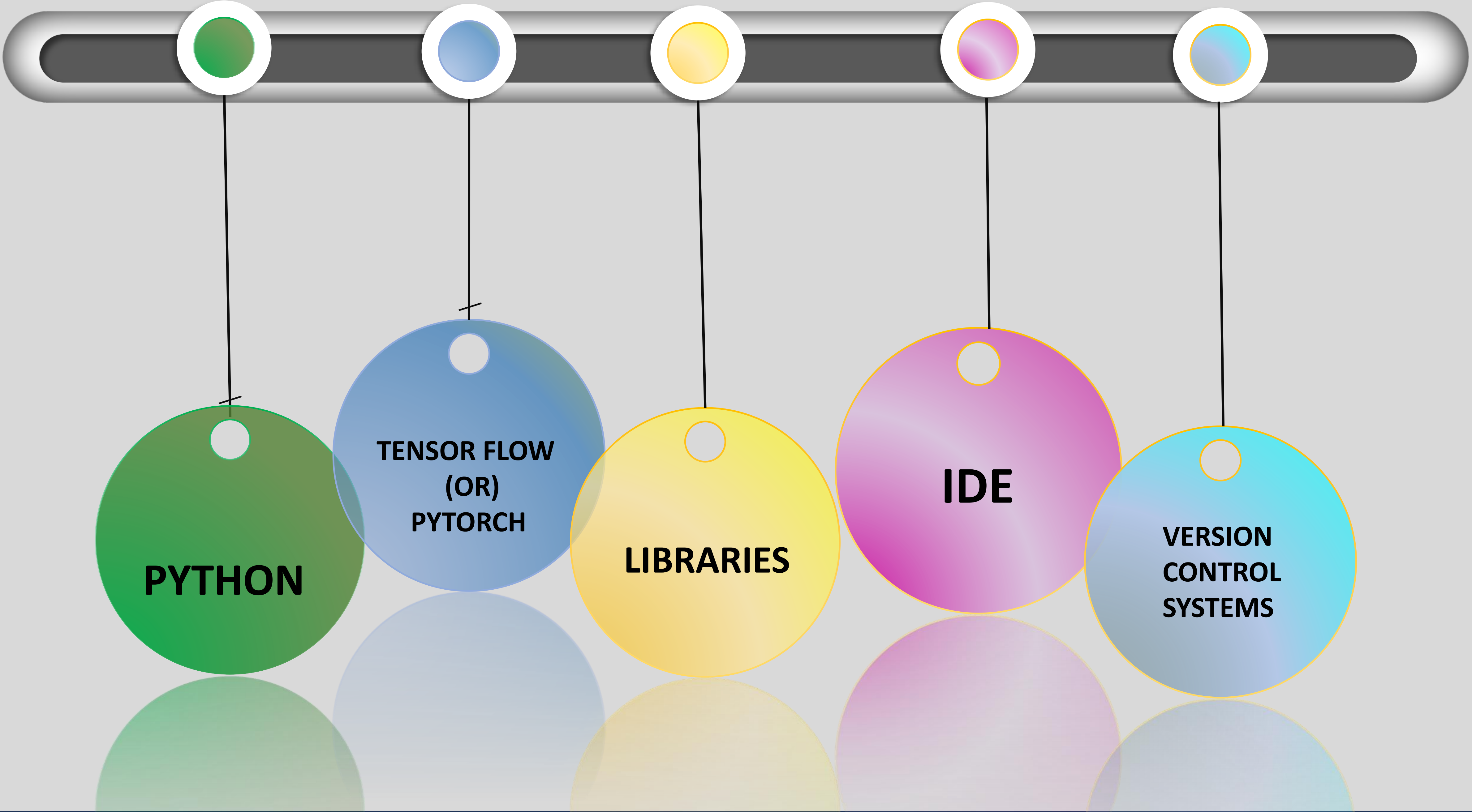
GPU (OPTIONAL)

RAM

STORAGE

INTERNET

SOFTWARE REQUIREMENTS



PYTHON

TENSOR FLOW (OR) PYTORCH

LIBRARIES

IDE

VERSION CONTROL SYSTEMS

SOFTWARE REQUIREMENT SPECIFICATION

- **Input:**
  - Upload Content and Style Images.
- **Processing:**
  - Pre-process Images (Resize and Normalize).
  - Feature Extraction (VGG-19).
  - Loss Computation (Content, Style, Total Variation).
  - Optimization (Gradient Descent or L-BFGS).
- **User Interaction:**
  - Simple image upload interface.
- **Technologies:**
  - TensorFlow with Keras.
  - VGG-19 Architecture.
- **Outcome:**
  - Display or Save Generated Image.

DESIGN CONCEPTS

1. Neural Network Architecture:

•Utilize a pre-trained neural network architecture, such as VGG-19, for effective feature extraction during style transfer.

2. Feature Extraction Layers:

•Identify specific layers within the chosen neural network for content and style representation, striking a balance between preserving content and transferring styles accurately.

3. Loss Functions:

•Implement content loss to ensure the preservation of essential content features.

•Style loss should effectively capture and transfer the artistic styles from the style image.

•Integrate total variation loss for smoother and more visually pleasing results.

4. Optimization Algorithm:

•Select an optimization algorithm suitable for your application, considering factors like convergence speed and resource efficiency. Options include Gradient Descent or L-BFGS.

Constraints

1. Computational Resources:

•Consider the computational demands of feature extraction and optimization, ensuring compatibility with a range of devices.

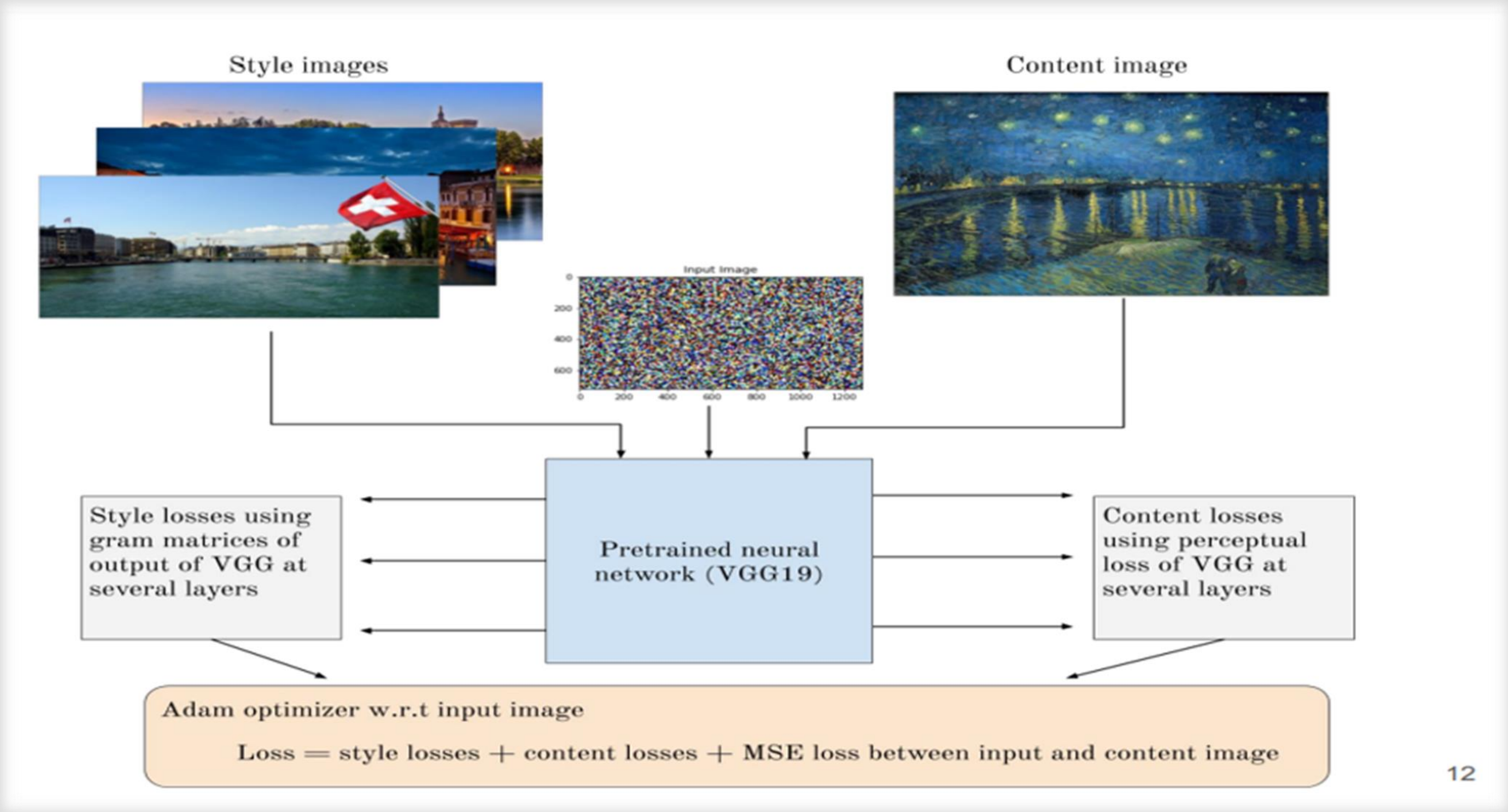
2. Processing Time:

•Optimize for reasonable processing times to provide a responsive and efficient user experience.

3. Memory Constraints:

•Be mindful of memory usage, especially when dealing with larger image files or running on devices with limited resources.

DESIGN DIAGRAM OF THE SYSTEM



Style losses using gram matrices of output of VGG at several layers

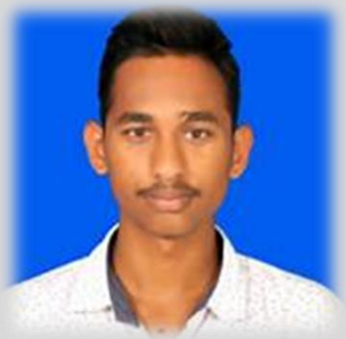
Content losses using perceptual loss of VGG at several layers

Adam optimizer w.r.t input image

Loss = style losses + content losses + MSE loss between input and content image

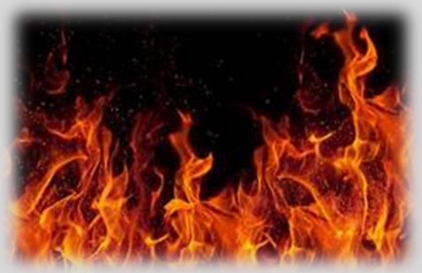
CONCEPTUAL DESIGN

1



CONTENT IMAGE

2

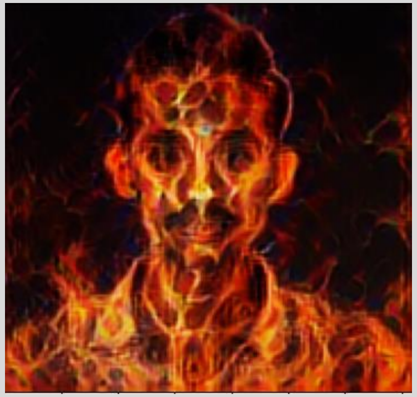


STYLE IMAGE

3

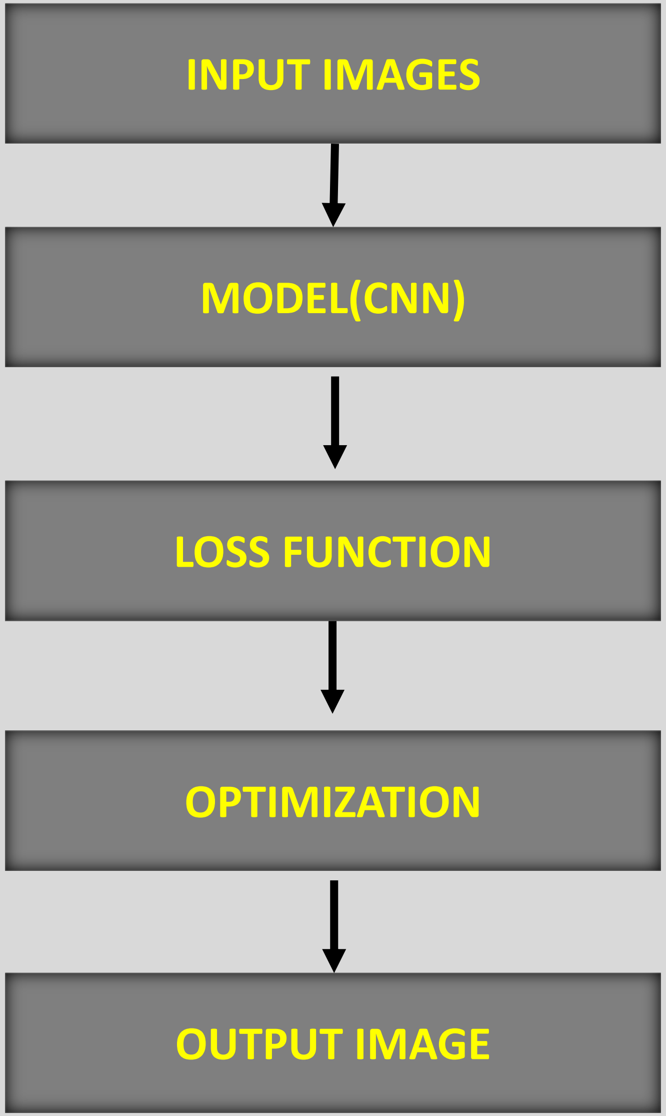
ALGORITHM

4



BLENDED IMAGE

LOGICAL DESIGN



INPUT IMAGES

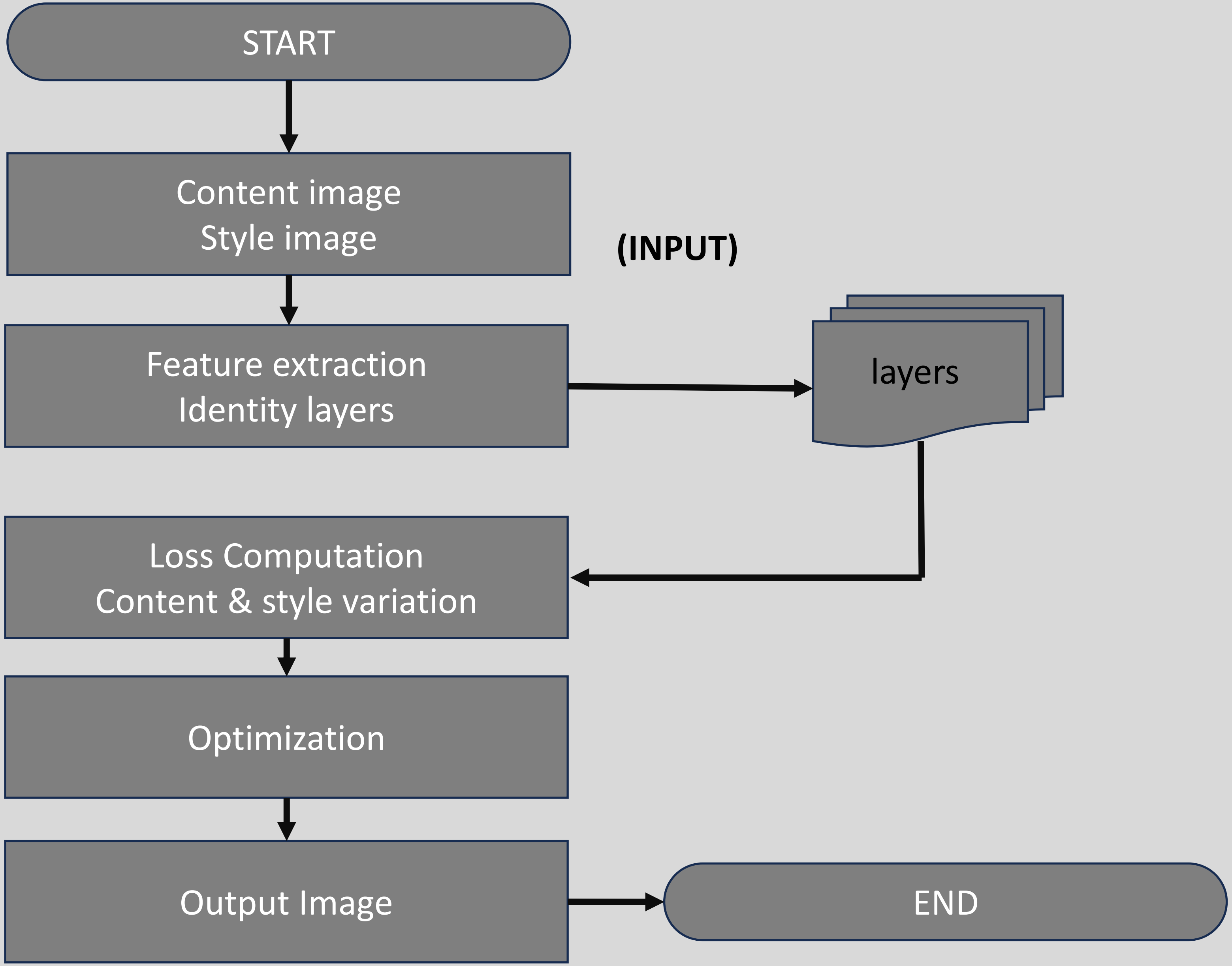
MODEL(CNN)

LOSS FUNCTION

OPTIMIZATION

OUTPUT IMAGE

ALGORITHM DESIGN



START

Content image
Style image

(INPUT)

Feature extraction
Identity layers

layers

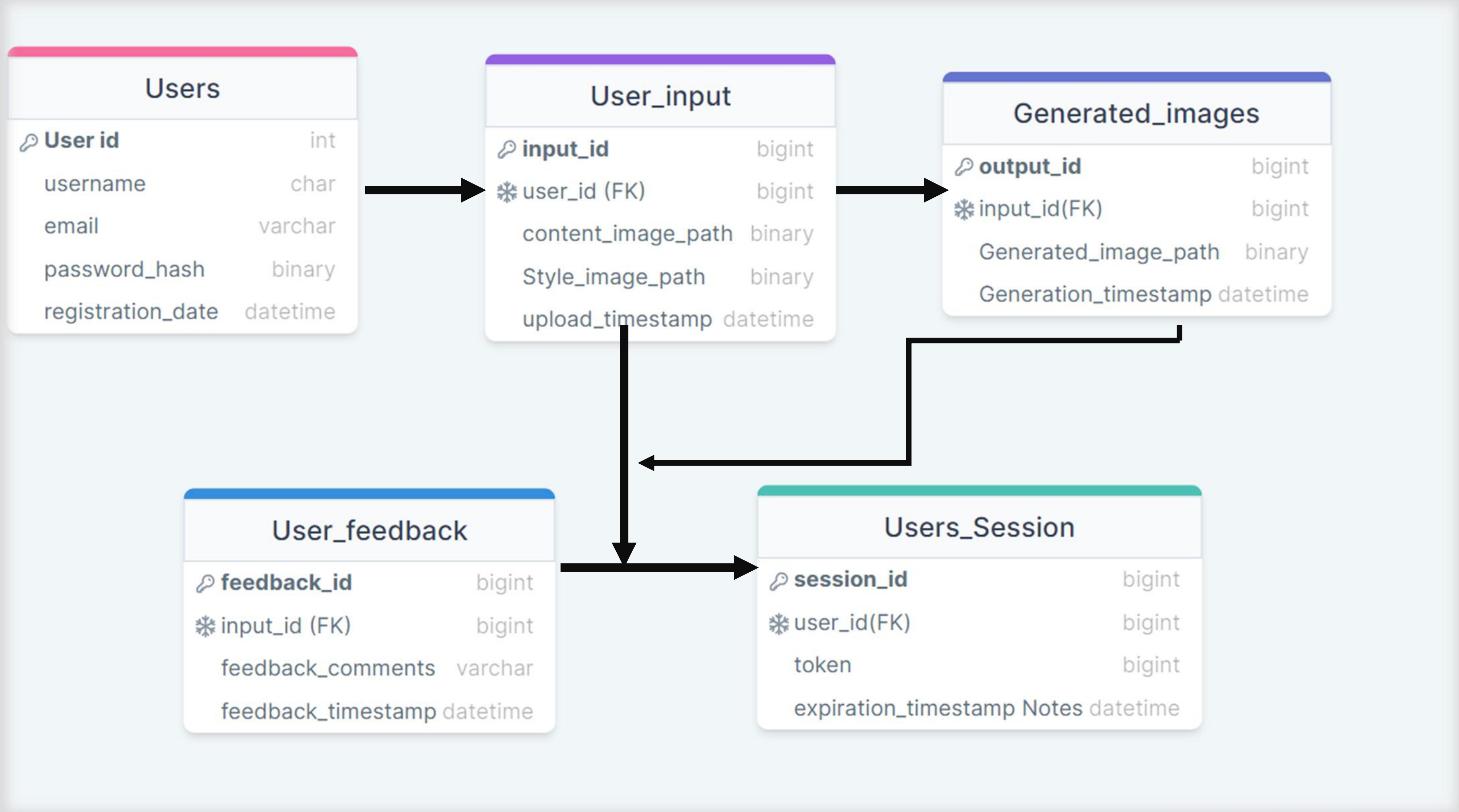
Loss Computation
Content & style variation

Optimization

Output Image

END

DATABASE DESIGN



Users	
User id	int
username	char
email	varchar
password_hash	binary
registration_date	datetime

User_input	
input_id	bigint
user_id (FK)	bigint
content_image_path	binary
Style_image_path	binary
upload_timestamp	datetime

Generated_images	
output_id	bigint
input_id(FK)	bigint
Generated_image_path	binary
Generation_timestamp	datetime

User_feedback	
feedback_id	bigint
input_id (FK)	bigint
feedback_comments	varchar
feedback_timestamp	datetime

Users_Session	
session_id	bigint
user_id(FK)	bigint
token	bigint
expiration_timestamp Notes	datetime

MODULE DESIGN SPECIFICATION

- Neural Network Module
- Input Processing Module
- Loss Computation Module
- Output Module
- User Interaction Module
- User Feedback Module
- Performance Optimization Module